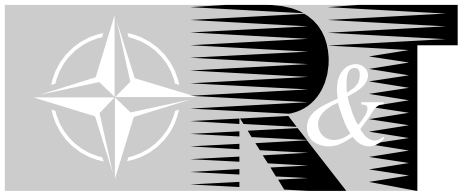
RTO-MP-101
AC/323(IST-033)TP/18

RTO-MP-101

NORTH ATLANTIC TREATY ORGANISATION



RESEARCH AND TECHNOLOGY ORGANISATION

BP 25, 7 RUE ANCELLE, F-92201 NEUILLY-SUR-SEINE CEDEX, FRANCE

© RTO/NATO 2003

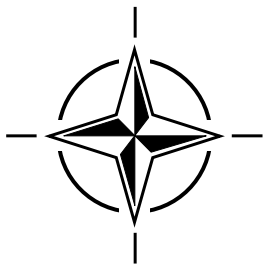
Single copies of this publication or of a part of it may be made for individual use only. The approval of the RTA Information Management and Systems Branch is required for more than one copy to be made or an extract included in another publication. Requests to do so should be sent to the address above.

RTO MEETING PROCEEDINGS 101

Real Time Intrusion Detection

(La détection des intrusions en temps réel)

Papers presented at the RTO Information Systems Technology Panel (IST) Symposium held in Estoril, Portugal, 27-28 May 2002.



Published June 2003

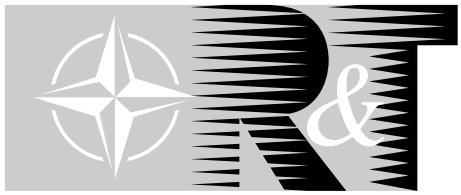
Distribution and Availability on Back Cover

This page has been deliberately left blank

Page intentionnellement blanche

RTO-MP-101
AC/323(IST-033)TP/18

NORTH ATLANTIC TREATY ORGANISATION



RESEARCH AND TECHNOLOGY ORGANISATION

BP 25, 7 RUE ANCELLE, F-92201 NEUILLY-SUR-SEINE CEDEX, FRANCE

RTO MEETING PROCEEDINGS 101

Real Time Intrusion Detection

(La détection des intrusions en temps réel)

Papers presented at the RTO Information Systems Technology Panel (IST) Symposium held in Estoril, Portugal, 27-28 May 2002.

The Research and Technology Organisation (RTO) of NATO

RTO is the single focus in NATO for Defence Research and Technology activities. Its mission is to conduct and promote cooperative research and information exchange. The objective is to support the development and effective use of national defence research and technology and to meet the military needs of the Alliance, to maintain a technological lead, and to provide advice to NATO and national decision makers. The RTO performs its mission with the support of an extensive network of national experts. It also ensures effective coordination with other NATO bodies involved in R&T activities.

RTO reports both to the Military Committee of NATO and to the Conference of National Armament Directors. It comprises a Research and Technology Board (RTB) as the highest level of national representation and the Research and Technology Agency (RTA), a dedicated staff with its headquarters in Neuilly, near Paris, France. In order to facilitate contacts with the military users and other NATO activities, a small part of the RTA staff is located in NATO Headquarters in Brussels. The Brussels staff also coordinates RTO's cooperation with nations in Middle and Eastern Europe, to which RTO attaches particular importance especially as working together in the field of research is one of the more promising areas of initial cooperation.

The total spectrum of R&T activities is covered by the following 7 bodies:

- AVT Applied Vehicle Technology Panel
- HFM Human Factors and Medicine Panel
- IST Information Systems Technology Panel
- NMSG NATO Modelling and Simulation Group
- SAS Studies, Analysis and Simulation Panel
- SCI Systems Concepts and Integration Panel
- SET Sensors and Electronics Technology Panel

These bodies are made up of national representatives as well as generally recognised 'world class' scientists. They also provide a communication link to military users and other NATO bodies. RTO's scientific and technological work is carried out by Technical Teams, created for specific activities and with a specific duration. Such Technical Teams can organise workshops, symposia, field trials, lecture series and training courses. An important function of these Technical Teams is to ensure the continuity of the expert networks.

RTO builds upon earlier cooperation in defence research and technology as set-up under the Advisory Group for Aerospace Research and Development (AGARD) and the Defence Research Group (DRG). AGARD and the DRG share common roots in that they were both established at the initiative of Dr Theodore von Kármán, a leading aerospace scientist, who early on recognised the importance of scientific support for the Allied Armed Forces. RTO is capitalising on these common roots in order to provide the Alliance and the NATO nations with a strong scientific and technological basis that will guarantee a solid base for the future.

The content of this publication has been reproduced
directly from material supplied by RTO or the authors.

Published June 2003

Copyright © RTO/NATO 2003
All Rights Reserved

ISBN 92-837-0032-5

Printed by St. Joseph Print Group Inc.
(A St. Joseph Corporation Company)
1165 Kenaston Street, Ottawa, Ontario, Canada K1G 6S1

Real Time Intrusion Detection

(RTO MP-101 / IST-033)

Executive Summary

Within NATO member nations and coalition partners there will be an increasing dependence on communication and information systems (CIS) to ensure the success of military operations, including mission critical operations. Also the interconnection of coalition CIS and the growing use of commercial-off-the-shelf software increases the risk of intrusions from external and internal sources. To minimize losses and ensure the continuous operation of CIS, there is a recognised need for a real-time, automated response to intrusions.

One of the important prerequisites for an appropriate response is the timely detection of intrusions, and this forms the background for the symposium.

The symposium includes two keynote addresses and seventeen papers discussing several aspects of the theme. The papers are presented in six technical sessions.

The first keynote address, entitled **Networked Systems Survivability Program**, is about the possibility of building survivable systems instead of continuing to correct the inadequacies of the systems being built today. The second keynote address, entitled **Building Secure Software**, is about taking security into account during all phases of development and ensuring that software developers get proper security training.

The first technical session, entitled **Real-time Intrusion Detection, Overview and Practical Experience** has 3 papers. They give an overview of the topics of the theme and point out some of the challenges of intrusion detection for the R&D community. In particular, practical experience illustrates the gap between actual needs and the state of intrusion detection systems.

The second technical session, entitled **Correlation and Fusion**, has 3 papers. They discuss technology for the correlation and fusion of intrusion detection information. The technology aims at faster and more reliable detection. One paper discusses fusion at the alert level.

The third technical session, entitled **Insider Threat Detection**, has 2 papers. The insider threat is a big challenge, because intrusion by authorized users may imply more severe consequences. Although several papers in the symposium deal with this aspect, the two papers selected for this session reflect the topic in specific environments.

The fourth technical session, entitled **Real-time Data Analysis and Processing** has 3 papers. It is obvious that a real time analysis of intrusion detection data is a very convenient way for real-time detection. The papers discuss anomaly detection techniques, clustering techniques, and data reduction techniques to increase speed of the analysis.

The fifth technical session entitled **Real Time Decision Support and Visualisation**, has 3 papers. The topics of this session are related to the fact that incident response will often include human decisions, thus intrusion detection systems must provide reliable information for decision making, e.g. appropriate visualisations of intrusions.

The sixth technical session, entitled **Intrusion Detection for Real Time and Time Service Dependent Applications**, has 3 papers. For real time applications, such as multimedia traffic and IP telephony, the detection of attacks on time dependency require special methods and technologies. Examples of real time applications are multimedia traffic and IP telephony. There is also a paper about a time-dependent service – thus attacks on time synchronisation can lead to unreliable service. Future intrusion detection systems must deal with this too.

The presentations at the symposium illustrate some R&D initiatives which will enable more reliable and timely detection of intrusions, but more improvements are still needed as basis for real-time automated response to intrusions. It is important for NATO to carefully monitor developments and apply the solutions so as to gain experience in military environments.

La détection des intrusions en temps réel

(RTO MP-101 / IST-033)

Synthèse

A l'avenir, la réussite des opérations militaires, y compris les opérations indispensables à la mission, conduites par les pays membres de l'OTAN et les partenaires d'une coalition, dépendra de plus en plus de l'efficacité des systèmes de communication et d'information (CIS). Aussi, l'interconnexion des systèmes CIS de coalition, associée au recours de plus en plus courant qui est fait à des composants du commerce (COTS), a pour effet d'accroître le risque d'intrusions de sources internes et externes. La nécessité de trouver une réponse automatisée aux intrusions, en temps réel, afin de réduire au minimum les pertes et d'assurer le fonctionnement ininterrompu des systèmes CIS, est largement admise.

Ce symposium sur la détection d'intrusions en temps réel a pour origine le sujet de la détection d'intrusions en temps réel, qui est l'une des conditions préalables à l'obtention d'une réponse appropriée.

Le symposium comprend deux discours d'ouverture et dix-sept communications couvrant différents aspects du sujet. Les communications sont présentées en six sessions techniques.

Le premier discours d'ouverture, intitulé **La réalisation de systèmes résistant en réseaux**, examine la possibilité de réaliser des systèmes résistants, pour ne pas avoir à corriger les imperfections des systèmes en cours de fabrication aujourd'hui. Le deuxième discours d'ouverture, intitulé **La création de logiciels protégés**, concerne la prise en compte de la sécurité pendant toutes les phases du développement, ainsi que la nécessité de formation en matière de sécurité pour les réalisateurs de logiciels.

La première session, intitulée **La détection d'intrusions en temps réel : aperçu et expérience**, comporte trois communications. Elles résument les différents sujets du thème, en faisant ressortir certains défis en matière de détection d'intrusions qui seraient à relever par les chercheurs dans ce domaine. En particulier, l'expérience pratique montre l'écart qui existe entre les besoins réels et les performances actuelles des systèmes de détection d'intrusions.

La deuxième session technique, intitulée **La corrélation et la fusion**, comporte trois communications. Elles portent sur les technologies de la corrélation et la fusion des données de détection d'intrusions. Ces technologies visent une détection plus rapide et plus fiable. L'une d'entre elles concerne la fusion au niveau de l'alerte.

La troisième session technique, intitulée **La détection de la menace interne**, comporte deux communications. La menace interne représente un défi important, parce que l'intrusion par des utilisateurs autorisés peut avoir des conséquences plus graves. Bien que cet aspect soit traité par d'autres conférenciers, les deux communications choisies pour cette session examinent le sujet dans le contexte d'environnements spécifiques.

La quatrième session technique, intitulée **L'analyse et le traitement de données en temps réel**, comporte trois communications. Il est un fait certain que l'analyse en temps réel des données de détection d'intrusions est très importante pour la détection en temps réel. Les communications traitent des techniques de détection d'anomalies, des techniques d'agrégation, ainsi que des techniques destinées à accélérer l'analyse.

La cinquième session technique, intitulée **La visualisation et les aides à la prise de décisions**, comporte trois communications. Les sujets couverts par cette session témoignent du fait que la réponse aux incidents implique souvent des décisions humaines et que par conséquent, les systèmes de détection d'intrusions doivent fournir des données fiables, aux fins de la prise de décisions, par exemple des visualisations d'intrusions.

La sixième session technique, intitulée **La détection des intrusions pour applications temps réel et asservies au temps**, comporte trois communications. Pour les applications temps réel, telles que le trafic multimédia, la détection d'attaques sur les fonctions liées au temps, fait appel à des méthodes et à des technologies particulières. Le trafic multimédia et le téléphone Internet sont des exemples d'applications multimédia. Une autre communication présente un service asservi au temps : dans de tels cas, la fiabilité du service peut être compromise par d'éventuelles attaques sur la synchronisation. Les futurs systèmes de détection d'intrusion devront pouvoir traiter cet aspect également.

Les présentations qui seront données lors du symposium renseigneront les participants sur certaines initiatives R&D qui devraient permettre d'obtenir une détection d'intrusions plus prompte et plus fiable, mais la réponse aux intrusions automatisée en temps réel ne sera obtenue que suite à d'autres améliorations encore. Il est important pour les pays membres de l'OTAN de suivre de près les développements dans ce domaine et d'appliquer les solutions trouvées afin d'acquérir de l'expérience dans des environnements militaires.

Click inside the blue boxes or on the titles to view the corresponding section

Contents

SESSION III: INSIDER THREAT DETECTION
Chairman: Dr H. STEINHEUER (GE)

SESSION IV: REAL TIME DATA ANALYSIS AND PROCESSING
Chairman: Dr J. LEFEBVRE (CA)

SESSION V: REAL TIME DECISION SUPPORT AND VISUALISATION
Chairman: Dr W. MEES (BE)

SESSION VI: INTRUSION DETECTION FOR REAL TIME AND TIME SERVICE DEPENDENT APPLICATIONS
Chairman: Prof. A. ZUQUETE (PO)

Theme

The extensive and increasing application and use of information and communication technology has created both new capabilities and new vulnerabilities. It is critical to the future of NATO that systems remain robust and retain the confidence of the users. The proliferation of threats to information systems has already been seen in the civil sector. These threats will also impact military operations, especially with the proliferation of commercial-off-the-shelf systems and technology. Probing of computer networks and penetration attacks have already been reported by NATO-member nations and are expected to increase. These threats originate not only from potential military adversaries and state-sponsored terrorists, but also from third parties whose only interest is to disrupt operations. Another problem may be the threat of internally originated attacks or misuse.

The widespread use of information systems for the collection, processing and dissemination of mission critical data and information requires appropriate defences. It is important to understand how to identify unauthorised activities in order to provide a real-time (or near-real-time) automated response to ensure the operation of information systems, which need to be continuously available, including mission critical systems. In this context, a key challenge is the development of systems for real-time detection of intrusions.

The main focus of this symposium is technologies/techniques and tools for real-time intrusion detection. This will include a state-of-the-art overview of research, experiments, and deployment of promising current and new intrusion detection technologies, which have the potential for real-time applications, thereby making it possible to see which technologies and tools could be deployed in the future within NATO and coalition networks.

TOPICS TO BE COVERED:

- Real-time decision support
- High-speed data processing, including pre-processing techniques, data reduction and adaptive filtering
- Advanced real-time data analysis
- Information fusion, including correlation, aggregation, and alignment techniques
- Insider threat detection and intruder profiling
- Expert systems or intelligent assistants for intrusion detection
- Pattern recognition, including Neural Nets (e.g. adaptive resonance techniques)
  On-line visualisation of intrusions and progress thereof

Thème

L'application et l'utilisation largement répandues et croissantes des technologies de l'information et des communications ont créé non seulement de nouvelles capacités mais aussi de nouvelles vulnérabilités. Il est indispensable pour l'avenir de l'OTAN que les systèmes conservent leur robustesse et continuent de mériter la confiance des utilisateurs. La prolifération des menaces sur les systèmes d'information a déjà été remarquée dans le secteur civil. Ces menaces auront également un impact sur les opérations militaires, qui sera amplifié par la prolifération des systèmes et technologies disponibles sur étagère (COTS). Le sondage des réseaux informatiques et des attaques de pénétration ont déjà été signalés par des pays membres de l'OTAN et deviendront sans doute plus nombreux. Ces menaces ont pour origine non seulement des adversaires militaires potentiels et le terrorisme parrainé par l'état, mais aussi des tiers dont le seul intérêt est de perturber les opérations. La menace d'attaques et d'usage abusif d'origine interne est un problème additionnel.

L'utilisation généralisée de systèmes d'information pour la collecte, le traitement et la diffusion de données indispensables à la mission exige des moyens de défense appropriés. Il est important de savoir identifier des activités non autorisées afin de fournir une réponse automatisée en temps réel (ou quasi-réel) pour secourir les systèmes d'information, qui doivent fonctionner en permanence, y compris les systèmes indispensables à la mission. Dans ce contexte, le développement de systèmes assurant la détection d'intrusion en temps réel est l'un des défis clés.

Ce symposium est principalement axé sur les technologies/techniques et outils de détection d'intrusion en temps réel. Le programme comprend un aperçu de l'état actuel des connaissances dans le domaine de la recherche, l'expérimentation et le déploiement des technologies de détection d'intrusion prometteuses actuelles et nouvelles, susceptibles d'être utilisées pour des applications temps réel, et laisse prévoir ainsi les technologies et les outils qui pourraient être déployés à l'avenir au sein des réseaux de l'OTAN et d'éventuelles coalitions.

SUJETS A TRAITER :

- Le soutien de la prise de décisions en temps réel
- Le traitement des données à grande vitesse, y compris les techniques du pré-traitement, la réduction des données et le filtrage adaptatif
- L'analyse des données avancée en temps réel
- La fusion des données, y compris les techniques de corrélation, d'agrégation et d'alignement
- La détection de menaces internes et l'établissement de profils d'intrusion
- Systèmes experts et assistants intelligents pour la détection d'intrusions
- Analyse typologique, y compris les réseaux neuronaux (par exemple les techniques de résonance adaptative)
  La visualisation en ligne d'intrusions et de leur progrès

Information Systems Technology Panel

Chairman

Dr M. VANT
Deputy Director General
Defence Research Establishment Ottawa
Dept of National Defence
3701 Carling Avenue
Ottawa, ON, K1A 0Z4
CANADA

Deputy Chairman

Dr R. JACQUART
Directeur du DTIM
ONERA/CERT/DTIM
BP 4025
31055 Toulouse Cedex 4
FRANCE

TECHNICAL PROGRAMME COMMITTEE

GENERAL CHAIRMAN:	Prof. A. MILLER	US
TECHNICAL CHAIRMAN:	Dr A. MOELLER	DE
MEMBERS:	Prof. W. MEES	BE
	Dr J. LEFEBVRE	CA
	Dr J. BYDZOVSKY	CZ
	Dr H. STEINHEUER	GE
	Prof. G.L. FORESTI	IT
	Dr E. LUIIJF	NE
	Lt. A. ARCIUCH	PL
	Prof. A. ZUQUETE	PO

PANEL EXECUTIVE

From Europe:
RTA-OTAN
Lt.Col. A. GOUAY, FAF
IST Executive
7 Rue Ancelle, BP 25
F-92201 Neuilly sur Seine, Cedex
FRANCE

Telephone: +33 (1) 5561 2280 / 82 - Telefax: +33 (1) 5561 2298 / 99

From the USA or CANADA:
RTA-NATO
Attention: IST Executive
PSC 116
APO AE 09777

HOST NATION LOCAL COORDINATOR
Capt. J.C. DA SILVA VERISSIMO
Ministry of Defence
Regimento de Transmissoes
Batalhao de Estruturas
Rua de Sapadores
1170 Lisboa, PORTUGAL

Acknowledgements/Remerciements

The Panel wishes to express its thanks to the Portuguese members of RTA for the invitation to hold this Symposium in Estoril and for the facilities and personnel which made the Symposium possible.

Le Panel tient à remercier les membres du RTB du Portugal auprès de la RTA de leur invitation à tenir cette réunion à Estoril, ainsi que pour les installations et le personnel mis à sa disposition.

Technical Evaluation Report

S.K. Dahel
Defence R&D Canada – Ottawa
3701 Carling Avenue
Ottawa, Ontario, Canada, K1A 0Z4

Introduction

This paper is the technical evaluation report for the NATO/RTO/IST symposium on Real-Time Intrusion Detection (RTID) held in Estoril, Portugal from May 27 to May 28 2002.

In this introductory section, we present some general information regarding the symposium including its arrangements and some statistics about the attendance. In the succeeding sections, we review the theme and the focus of this symposium. This is followed by a technical overview of the symposium's presentations and an evaluation and analysis of the papers/presentations against the objectives/purpose of the symposium. Finally, we provide an overall conclusion on the symposium, highlighting trends; making suggestions/recommendations for future activities; and providing additional miscellaneous remarks.

The symposium was held at the Estoril Eden Hotel just outside Lisbon, Portugal. Most of the attendees and presenters stayed in the hotel ensuring a high attendance for all of the two-day symposium's presentations. There were 117 participants, representing all but two NATO countries and four Partnership for Peace (PfP) nations.

The welcoming address was given by Dr Manuel da Cunha Rego, Director of MOD Infrastructures General Directorate (Portugal). Following this, Dr. M. Vant, the IST Panel Chairperson, gave an overview on the NATO/RTO/IST panel. Prof. A. Miller, the Symposium Chairperson, gave the introductory remarks.

The Technical programme consisted of two keynote addresses and seventeen technical presentations from six different nations. Of all the contributions, ten were from academia and five from Defence/Government organizations.

Theme and Focus of the Symposium

We begin by examining the rationale that led to the convening of this symposium and we review its intended purpose.

The increasing reliance on information systems has coincided with an increase in the number of threats against both civilian and military information systems, especially with the growing dependence on commercial-off-the-shelf (COTS) systems. Several organizations and countries, including NATO-member nations, have reported an upward trend in malicious/unauthorised network activities. These activities include network probes and penetration attacks. In addition to these external threats, internal threats exacerbate the problem. Some of these systems are mission critical and may need to be constantly available. To ensure a continuous operation of these information systems, it is essential to understand how to identify unauthorised activities to be able to react/respond automatically in real-time or near-real-time (RT or NRT). A key challenge, as recognised by this symposium's organisers, is the development of systems for real-time detection of intrusions.

The purpose of this symposium was "the exchange information on the state-of the-art and state-of-practice in real-time intrusion detection". The focus, as stated in the symposium's announcement, was on technologies, techniques and tools for real-time intrusion detection. Contributions were expected in areas that included state-of-the-art overviews of research, experiments, and deployment of promising current and new intrusion detection technologies that have a potential for real-time applications. Specifically, the advertised topics included:

- real-time decision support;
- high-speed data processing (including pre-processing techniques, data reduction and adaptive filtering);
- advanced real-time data analysis; Information fusion(e.g. correlation, aggregation, and alignment techniques);
- insider threat detection and intruder profiling;
- expert systems or intelligent assistants for intrusion detection;
- pattern recognition (e.g. Neural Nets, adaptive resonance techniques); and
- on-line visualization of intrusions and the progress thereof.

It was hoped that this would allow participants to see which technologies and tools could be deployed in the future within NATO and coalition networks.

There were two keynote talks and six technical sessions. The sessions consisted of seventeen presentations and were organised along the following themes:

SESSION I	Real-Time Intrusion Detection, Overview and Practical Experience;
SESSION II	Correlation And Fusion
SESSION III	Insider Threat Detection
SESSION IV	Real-Time Data Analysis and Processing
SESSION V	Real-Time Decision Support and Visualization
SESSION VI	Intrusion Detection for Real-Time and Time-Service Dependent Applications

Technical Content

In this section, we provide a brief overview of the symposium's technical content as delivered in the presentations. We first examine the two keynote presentations followed by each of the six technical sessions.

Keynote Presentations

The first keynote address, entitled "Building Survivable Systems - New Challenges in the New IT Environment", was delivered by Mr Richard Pethia, Director of the Networked Systems Survivability Program at the Software Engineering Institute at Carnegie Mellon University. In his keynote, Mr.Pethia reminded us of how dependent we have become on large-scale, highly-distributed systems. He then described the different types of computer incidents and attacks and their trends. The major trends include the increased automation and speed of attack tools; the increased sophistication of attack tools; the faster discovery of vulnerabilities; the increasing permeability of firewalls; the increasing asymmetric threat; and the increasing threat from infrastructure attacks. Consequently, he outlined some interesting perspectives on Information Technology (IT) shifts: Central to Global; Bounded to Unbounded; Insular to Networked; Predictable to Asynchronous; Single Responsibility to Shared; and Overhead to Essential. He argued that yesterday's solutions will not work in today's environment and proposed a shift in the approach from security to survivability. He suggested shifting from security, as a narrow technical specialty accessible only to experts and dealing with the protection of specific components, to survivability, as a risk management problem requiring the involvement of the whole organization to support the survival of the organization's mission.

The second keynote address, entitled "Building Secure Software," was delivered by Dr Gary McGraw, Chief Technology Officer at Digital Inc., Dulles, Virginia. His presentation was lively and, some may say, provocative. It helped to generate many informal discussions among participants outside of the normal programme. He stressed the importance of building secure software at all stages, especially during design, as opposed to today's "defence-in–depth" paradigm of retrofitting software for security. He presented some examples and gave anecdotes about common bugs in today's software. He emphasised the importance of proper security training for software developers in addition to the traditional focus on those responsible for network administration.

Technical Sessions

Session I - Real-Time Intrusion Detection, Overview and Practical Experience

There were three presentations in this session. The first presentation (Luiijf and Coolen) covered an overview of the different types of intrusion detection systems (IDS), including a generic model, functionalitiy and components of an IDS. Perhaps one of the most relevant aspects of this session was the time-axis-model showing the progress of an attack starting with the reconnaissance phase (pre-attack) and continuing to the possible damage phase (post-attack). Using this model, the necessity for real-time intrusion detection was highlighted. In the other presentations (Šimon and Triznová; Coolen, Vand Geloven and Luiijf) the presenters shared their personal experiences and the lessons learned in using and deploying IDSs.

Session II - Correlation and Fusion

Three presentations were delivered in this session. The first presentation (Wolthusen) was about a policy-based security architecture that can be retrofitted onto COTS operating systems such as Unix- and Microsoft Windows NT. This enables the use of individual hosts as both IDS data collection and fusion sites. As part of the MIRADOR project, sponsored by the French DGA/CELAR/CASSI, the second presentation dealt with an approach to develop a cooperation module for analysing alerts and generating more global and synthetic alerts. The project's approach is based on an "explicit" correlation of events where a security administrator can express some connection among the events, be it through logical or topological links. In particular, the key objective is to correlate alerts from elementary attacks, which corresponded to the steps of much larger attacks, and thus recognise and identify the overall intrusion plan executed by the intruder. These attacks are specified using an XML-based language known as LAMBDA. The final presentation in this session (McCallam,Whitson and Zavidniak) focused on the correlation and fusion of events and information. The presenter first reminded us that there is no real difference between an inside attacker and an outside attacker except that a 'true" insider is always intentionally malicious. In addition to cyber events and timeline, he stressed the use of other non-traditional factors, such as physical location, in the fusion process. To illustrate this, he used a timeline chart similar to the one presented in Session I's first presentation with the addition of some prior knowledge about a physical location. In particular, he showed how each individual access to part of the physical location was insufficient to draw wide conclusions, but when taken together they can provide significant clues.

Session III - Insider Threat Detection

Two presentations were given in this session. The first presentation (Toelle and Plaggemeier) addressed the problem of monitoring encrypted traffic. It proposed a prototype to deal with traffic tunnelled through a secure shell connection (SSH). The basic idea of the approach is to use a transparent proxy. The proxy accepts connection requests from SSH clients and forwards data to the selected SSH-server. Because some anomaly and misuse detection are integrated into this proxy, it is possible to have an "active" IDS where traffic is screened and checked against a database before being forwarded, if it passes the screening, or being dropped, if an attack is discovered. The anomaly detection is based on a "sequence tree" technique. The second presentation (Valvis, Sklavos and Polemi) outlined a solution that attempts to integrate a mission-critical legacy system securely into a modern e-business environment (three-tier architecture). The proposed

solution concentrates on the mitigation of misuse threats coming from a trusted insider. The solution uses a modular design, is based on a security policy, and includes misuse detection functionality.

Session IV - Real-Time Data Analysis and Processing

This session was composed of three presentations. The first presentation (Büschkes, Seipold and Wienzek) discussed the use of so-called transaction-based anomaly detection techniques. The authors claim that, unlike conventional anomaly detection techniques, which often require complex and time consuming computations (e.g. multivariate statistics), transaction-based anomaly detection techniques are not computationally complex. In these techniques, the expected protocol or application behaviour is modelled by extended finite state machines. They examined the performance of these techniques, for real-time monitoring of communication networks, from both theoretical and practical viewpoints. Subject to several optimizations, they concluded that a 100 Mbit/s communication link, involving up to four protocol layers and using a connection-oriented protocol at the transport layer (TCP), can be monitored using these techniques. In the second presentation (MacLeod and Whyte), the impact of partial IP datagram inspection on signature-based network IDSs was examined. By reducing how much of the IP datagram gets inspected and analysed, signature-based network IDSs are able to increase throughput and minimise the amount of incident data storage. However, this improvement is not cost free since a significant number of attacks may still be missed and a more generic alarm may be issued instead of instead of a more specific alarm signaling an intrusion.. It was advised not to use a standard capture length to inspect protocol headers because various protocols have different header lengths. The third presentation (Skormin, Summerville, Moronski and Sidoran) discussed an approach, utilizing a statistical analysis of the data traffic through the network, that would enhance the administrator's ability to detect a distributed attack as early as possible. This approach is based on data mining and uses a modified cluster analysis aided by genetic optimization. It sets "correlation ellipses" in informative subspaces and uses them as separating rules. They have implemented and demonstrated this approach in a lab environment.

Session V - Real-Time Decision Support and Visualization

This session had three presentations. The first presentation (Holz, Meier and Koenig) considered the detection speed aspect for real-time applications, especially for host-based audit analysis. Their proposed solution to tackle this problem consists of a distributed intrusion detection infrastructure known as HEIDI, which provides a module system based on sensors and agents. According to the authors, HEIDI's basic features are distributed analysis functionality, handling of overload situations, and dynamic configurability. In addition, they claim to offset the problem of time-consuming audit analysis with the integration of "StraFER", a new signature match algorithm. A prototype to implement this solution is foreseeable in the near future. The second presentation (Gombault, and Diop) was a follow on to the second presentation from Session II. It discussed the reaction aspect in the MIRADOR project. They first proposed an action taxonomy and then showed how this may be used to provide a reliable and flexible reaction functionality that is easily integrated within the MIRADOR architecture. They indicated that real-time reaction capability is not widely used to avoid side-effect "Denial Of Service" attacks. The third presentation (Massicotte, Whalen and Bilodeau) was on a prototype network-mapping tool. This visual tool can be used to offer network and security administrators an extensive real-time picture of their network topology. It can provide descriptions for both physical and logical connectivity of network elements. It also identifies the operating systems running on the networked machines, indicates the state and shows configuration information about the hosts and their connectivity. To query network components, the tool uses built-in standard protocols and applications (ICMP, ARP, NetBIOS, DNS, SNMP); freely-available mapping tools (nmap, Xprobe); and a number of in-house developed tools for analysing the query results.

Session VI - Intrusion Detection for Real-Time and Time-Service Dependent Applications

There were three presentations in this last session. The first presentation (Wienzek, Büschkes and Borning) discussed possible attacks against multimedia streams and underlying protocols and architectures. They

described these attacks by modeling the behaviour of data sources and networks. The attacks were detected with an anomaly approach by observing deviations from the model. In addition, they carried out a performance analysis of this approach. The second presentation (Downard and Miller) addressed Kerberos vulnerabilities using IDS. The authors identified two major vulnerabilities but pointed out that these vulnerabilities have not yet caused widespread problems. In the last presentation (Steinebach et al.), the contributors examined the security of IP telephony. They proposed a combination of classic IDS and audio watermarking to enhance the user and data authentication, and to monitor calls.

Technical Evaluation

To set the stage and put things into context, we will first briefly review some key concepts of RT systems. We then analyse and discuss the threads of ideas, some of the key factors and promising technologies presented that may have impact on real-time intrusion detection (RTID), and IT security in general. Following this, we assess the relevance of the presentations by indicating which topics they covered and providing our own explanation of why real-time, the symposium's focus, was not as well-represented as some may have expected it to be. Finally, we present a summary of the participant's assessment based on the standard questionnaire collected during the symposium.

Traditionally, real-time computing and communication has been restricted to special-purpose systems (e.g. command-and-control systems, process control, and avionics). These embedded and mission critical systems have been designed with typical timing constraints. A common misunderstanding is that RT computing is synonymous with "fast" computing. However, it is generally accepted that real-time computing systems are characterised by, not just the logical results of the computation, but also by the time at which results are available. RT tasks have generally start/finish, hard/soft deadlines. RT system requirements usually include determinism, responsiveness, user control, reliability, and fail-soft operation. Recently, the realm of RT computing has extended beyond its usual single application control in a predictable environment (e.g. special operating systems and hardware) to non-predictable and under multi-application control (Web/Internet) where traditional RT techniques may not be applicable. In this environment, a symposium was a good forum to exchange ideas and information about RTID issues and available/potential solutions.

The different presentations covered several areas. Intrusion Detection was the focus but IT security in general was the main common thread. The two keynote talks shared the *building* idea of "survivable systems" and "secure software" respectively. They undoubtedly generated interesting and lively hallway discussions. Both keynote speakers shared a common view that a shift in today's approaches is required and proposed different courses. The first keynote speaker suggested an approach where the emphasis is on the ability of a system to fulfill its mission. He proposed three characteristics for a survivable system: resistance, the ability of a system to deter attacks; recognition, the ability to recognize attacks and the extent of damage; and recovery, the ability to restore services in a timely manner. In addition to securing the platform and the network, the second keynote speaker opted for attacking the problem "at the source" by building secure software. Initially, it may not be easy to see how these two approaches relate. They could even be seen as incompatible, especially considering the second keynote speaker did not advocate the "fortress model" and had reservations with the "defence-in-depth" practice, both of which use IDSs as important components. However, a closer look at the first keynote presenter will reveal that he too shared some of these opinions. For instance, his suggestion for a shift from security to survivability included a reconsideration of the fortress model. Despite their non-trivial relation to RTID, these two keynote presentations have provided the audience with many insights in the bigger picture of IT security.

Although the symposium's focus was expected to be on RTID, several contributions covered techniques and tools for IDS in different applications and environments ranging from traditional wired networks to wireless and IP telephony with only a few of them indirectly highlighting the RT aspects. A number of presentations focused on the ever-daunting problem of false alarms that plague most, if not all, IDSs. Some promising techniques tackling this problem, including fusion and aggregation, were delivered at this symposium. Other

presentations addressed the growing limitations of network-based IDSs, especially their blindness to encrypted traffic and their inability to cope with larger and larger pipes, particularly for real-time applications. We have noticed a recent surge in research activities related to host-based ID to address these limitations. The integration of host-based and anomaly detection are expected to be widely available in future commercial IDSs.

Table 1 below shows an analysis of the presentations. We assess the relevance of the presentations by indicating which of the advertised topics the presentation covered. A check (✔) indicates what we believe was the contribution's main topic while an (X) shows the minor topics. As depicted in the table, there were many contributions where the main topic did not match those advertised for the symposium. In particular, the real-time aspect was mostly cosmetic for some of the contributors.

Advertised Topics \ Presentations	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
Real-time decision support	X			X	X				X	X	X	X	✔				X
High-speed data processing, including pre-processing techniques, data reduction and adaptive filtering									X	X				✔	X		
Advanced real-time data analysis							X		✔		✔		X	X			✔
Information fusion, including correlation, aggregation, and alignment techniques				✔	✔	X							X				
Insider threat detection and intruder profiling				X		✔		X									
Expert systems or intelligent assistants for intrusion detection					X												
Pattern recognition, including Neural Nets (e.g. adaptive resonance techniques)							X										
On-line visualization of intrusions and progress thereof					X							X	X				
Others	✔	✔	✔				✔	✔		✔		✔	X		✔	✔	X

Table 1: Matching Presentations and Advertised Topics

As stated earlier, it seems that RT did not get as much attention as one may have perhaps expected. The limited number of presentations that dealt directly with RTID themes may indicate that RT is not an important issue for ID. Some may raise the question, "Was the symposium title appropriate/relevant?" Although some contributions alluded to time constraints, an important factor in RT applications, many of the contributions focused only on the ID part of RTID. This was possibly due to the nature of IDSs, which, until recently, have been passive systems and have typically run on top of operating systems where RT, if considered at all, was not an essential design factor. Perhaps "fast (real-time) reaction" should have been the focus of the symposium instead of "real-time detection." Though, we should point out that presentation 13, "MIRADOR: reaction function", addressed partially this topic. In all fairness, until recently "reaction" has not been part of IDS's functionality. Above all and as stated previously, traditional RT techniques and methods may not be applicable and new approaches should have been considered.

Let us look at the results from the standard questionnaires distributed at the symposium. Thirty-one participants filled out the questionnaire. From question 6, Paper 17 "IDS for IP telephony networks" (Steinebach et al.) was the most interesting paper. In addition, the attendees appreciated papers 2, 5 and 6. Almost unanimously, the audience found that the time allowed for presentation, discussion and exchange of ideas was about right. Question 13, on the quality of the translation, received the least favourable result.

About half of those who responded thought that quality was bad. Finally, the vast majority (71% of the respondents) found the symposium significant or extremely valuable to them and their organization.

Conclusions and Recommendations

We first review the purpose of the symposium and discuss its outcome. We then highlight some of the major trends as delivered by the contributions; along with some suggested topics for future activities. We will wrap up this section with some miscellaneous concluding observations and a recommendation for an exploratory team or a task group.

A symposium is usually a good forum to exchange information and explore collaboration possibilities in immature/new fields such as RTID. This symposium was successful and has met its purpose, "the exchange information on the state-of the-art and state-of-practice in real-time intrusion detection." Though the RT aspect was not as comprehensive as some may have expected, nevertheless, this symposium has allowed the NATO community to exchange knowledge about many on-going security-related projects and intrusion detection in particular. In fact, the vast majority of those who responded to the questionnaire have indicated that the symposium was significantly or extremely valuable to them and their organization.

Unlike other traditional defence topics, the ID area is still mostly commercially driven and the symposium has identified several trends in this field. They include the popularity of anomaly detection techniques as they are increasingly implemented in prototypes and COTS; the development of hardware-based ID techniques to meet the performance challenges and the time constraints of RTID; the usage of IDS in non-conventional environment (e.g. IP telephony; securing other security applications); and though still in its infancy, the convergence/integration of "active" (e.g. firewall) and "passive" security devices/systems (e.g. IDS).

In addition to using fusion and correlation techniques to reduce the rate of false alarm and enhance the detection capabilities, the symposium has identified several avenues for further investigation. The insider threat, as its timeline is much shorter than its external counterpart, seems to be a high potential threat for some of the contributors. In addition to authentication and non-repudiation of evidence gathering, the restoration of services in a timely manner was also viewed as an important future activity. These last two topics will be the subject of an upcoming IST/TGIA workshop in October 2002. Though, perhaps not technical, the balancing act of privacy versus security should continue to dominate policy-side activities. Automatic, safe and reliable real-time reaction has not yet been seriously addressed because of its possible undesirable side-effects (e.g. self-inflicted denial–of-service attacks). The integration of ID technologies, into perimeter devices and wireless networks without affecting their performance, is still in its early stages. Finally, visualization techniques and tools to deal with the info-glut generated by today's IDSs are still a long way away from materializing.

The following miscellaneous remarks are worth reiterating:

- While some initiatives are emerging to improve the security for IDSs or any other systems for that matter, their pace remains slow. As one keynote speaker pointed out, there is a shift from software being designed to meet only specific functional requirements, where security is an afterthought, to good software design/implementation, where security must be considered throughout the entire lifecycle of a system.
- With the inevitable continuing usage of commercial technology for military applications, it is important for the defence community to recognize the need of protection against more elaborated and sophisticated attacks that are not usually of commercial concern (e.g. covert channel).
- Without compromising national sovereignty issues, real-time sharing of intrusion and attack related information among coalition countries will be crucial to help thwart future threats. A common issue to almost all coalition operations is an architecture that allows such sharing.

This last remark deserves a closer attention. In addition to the topics for further investigation suggested above in this section, the problem of information sharing, although not unique to ID, is a topic of crucial importance. This problem is complicated by the sensitive nature of the information that needs to be shared. Unless properly addressed, the lack of prompt sharing could hinder future coalition operations. Given that "architecture and enabling technologies" is of interest to the IST panel and in light of recent technological developments, it is recommended that the panel consider setting up an exploratory team or a task group to deal solely with this important issue. Of particular importance is the secure sharing of information when conjugated with national sovereignty issues.

Introduction and Welcome

Dr. Malcolm R. Vant
Deputy Director General,
Defence R&D Canada-Ottawa
3701 Carling Avenue
Ottawa, Ont. K1A 0Z4
Canada

Click here to view PowerPoint presentation; Press Esc to exit

Prof. Ann Miller
Distinguished Professor of Electrical and Computer Engineering
University of Missouri-Rolla
125 Emerson Electric Co. Hall
Rolla, MO 65409-0040
United States

Click here to view PowerPoint presentation; Press Esc to exit

In the interests of readability and understandability, it is RTO policy to publish PowerPoint presentations only when accompanied by supporting text. There are instances however, when the provision of such supporting text is not possible hence at the time of publishing, no accompanying text was available for the following PowerPoint presentation.

Paper presented at the RTO IST Symposium on "Real Time Intrusion Detection", held in Estoril, Portugal from 27-28 May 2002, and published in RTO-MP-101.

This page has been deliberately left blank

Page intentionnellement blanche

Networked Systems Survivability Program

Richard Pethia
Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213-3890
USA

In the interests of readability and understandability, it is RTO policy to publish PowerPoint presentations only when accompanied by supporting text. There are instances however, when the provision of such supporting text is not possible hence at the time of publishing, no accompanying text was available for the following PowerPoint presentation.

Click here to view PowerPoint presentation; Press Esc to exit

Paper presented at the RTO IST Symposium on "Real Time Intrusion Detection", held in Estoril, Portugal from 27-28 May 2002, and published in RTO-MP-101.

This page has been deliberately left blank

Page intentionnellement blanche

Click here to view PowerPoint presentation; Press Esc to exit

Intrusion Detection Introduction and Generics

H.A.M. Luiijf, R. Coolen
TNO Physics and Electronics Laboratory
P.O. Box 96864
2509 JG The Hague, The Netherlands

E-mail: *luiijf@fel.tno.nl, coolen@fel.tno.nl*

Abstract

This paper gives an introduction to intrusion detection systems (IDSs) in order for the general audience to understand the specific R&D aspects discussed by the other symposium papers. The generic model of an IDS presented in this paper gives a common viewpoint for the study and discussion of functionality and components of IDS in a product independent way.
Furthermore, the necessity for real-time intrusion detection is highlighted using Logicon's time-axis model of an attack.

Introduction

Increasingly, NATO Forces operate in multinational coalitions and connect the NATO (nation) networks to non-NATO nations (NNNs), non-NATO international organisations (NNIOs), and non-governmental organisations (NGOs). Increasingly, operational requirements demand the sharing of information and integrating communication and information systems (CISs) of NATO nations and/or other coalition Forces. The use of interconnected modern information and communication technologies enhance the situational awareness and the strive for information dominance.
Internal and external threats to CIS, amplified by interconnecting with CIS of other nations and organisations, require early and often real-time warnings about intrusions and other irregularities in the NATO CIS as well as effective counter-measures. This to reduce the risks associated with potential unauthorised access to, compromise of, and control over NATO information and that of its members.
Intrusion Detection Systems (IDSs) are technical means that focus on the detection type of measures against intrusions in and to a CIS.

The Task Group on Information Assurance under the Information Systems Technology Panel of the NATO Research & Technology Organisation (NATO/RTO/IST/TG03) investigated the generics of Intrusion Detection Systems in 2000 – 2001 and published a report [9]. This paper highlights the main models and issues from that report as a foundation for this Real-time Intrusion Detection symposium.

Terms and definitions

In literature, different terms and definitions are used for intrusion and intrusion detection. In this report definitions are chosen in accordance with [1] and [2], but with a focus on the military operational CIS environment:

Attack: A deliberate intrusion in a CIS.

Attacker: The person, group, organisation or state that performs an attack.

Defender: The person, group or organisation (i.e. NATO agency or NATO nation) that is responsible for the CIS to be protected.

Intruder: The person, group, organisation or state responsible for an intrusion.

Intrusion: A deliberate or accidental unauthorised access to, activity against, and/or activity in, a CIS.

Intrusion Detection: The process of identifying that an intrusion has been attempted, will occur, is occurring, or has occurred.

Target: The CIS that an intrusion is aimed at.

Paper presented at the RTO IST Symposium on "Real Time Intrusion Detection", held in Estoril, Portugal from 27-28 May 2002, and published in RTO-MP-101.

Note that the definition of an intrusion includes intrusions that have an intentional or unintentional intend, harmful or harmless consequences, and concern both intrusions by insiders and outsiders (by definition respectively affiliated to the defender organisation or not).

Both unintentional intrusions and attacks can result in damage. This damage can concern the availability, integrity and/or confidentiality of the CIS.

Security measures

In order to see where IDSs fit in to the overall range of security measures, intrusion detection is positioned as one of the coherent security measures against an incident (e.g. an intrusion). The measures are presented using the security incident cycle, which is visualised in figure 1.

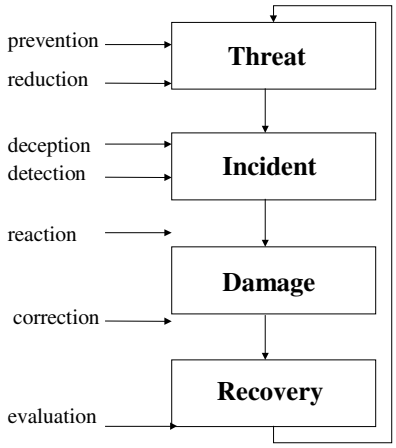


Figure 1: Security incident cycle

The incident part of the cycle consists of four elements: the threat, the incident, the occurrence of damage, and the recovery. The following different types of security measures are related to these elements: prevention, reduction, deception, detection, reaction, correction, and evaluation.

The security incident cycle has to deal with threats to the confidentiality, integrity and availability of the CIS. A defender first of all takes *prevention measures*. These measures prevent a threat from becoming a reality. An example of a prevention measure to protect an internal network is a Boundary Protection Device (BPD). Another example of a prevention measure is scanning for known vulnerabilities in a CIS and thereafter correcting these vulnerabilities by implementing patches or changing configuration parameters.
Reduction measures are measures that are performed in advance to reduce possible damage of an incident such as an intrusion. Examples of reduction measures are redundant systems, limitation of bandwidth, and regular back-ups.
Deception measures are a special type of security measures. They have the purpose to give false information to intruders, to reduce the possibility of an incident, to allow easier detection of an incident, to slow intruders down, or to obtain operational benefits over the intruding party.

Prevention, reduction and deception measures reduce the probability and the impact of an incident. However, this does not exclude possible occurrence. Therefore, the defender takes *detection measures*.
All intrusions have to be detected as early as possible. In this way, the defender does not loose valuable time over the intruder. This time can be used to identify the intruder and to take more extensive prevention, reduction, and deception measures to minimise damage and to maximise the possibility for a proper reaction. Intrusion detection is the main focus of IDSs. However, there is a tendency that other security measures such as reaction and deception are also incorporated in IDS-products.

After an intrusion is detected, the defender takes *reaction measures*. These reaction measures can be repressive in order to block the repetition of the intrusion. The reaction measures can also include tracing an intruder. Furthermore, if the operational authority for the CIS decides to start a process to press charges

against an intruder/attacker, inforensic[1] evidence often needs to be collected between the moment of the first intrusion related events, a successful intrusion in, and the recovery of the CIS [2].

When an intrusion results in damage to the integrity or availability of information, the next step in the security incident cycle is to take *correction measures* to undo at best the damage that was done. The operational status of vital parts of the CIS has to be reconstituted as soon as possible. This is where *reduction measures* such as back-ups prove their usefulness.

The final step in the security-incident cycle consists of an effectiveness *evaluation* of the security measures taken. Questions might be: 'what went well and what went wrong?' And what lessons can be learned and how to prevent a reoccurrence of the intrusion in the future?

Note that prevention, reduction, and detection measures should be designed according to the defence-in-depth principle [3]. That is the attacker or intruder should have to overcome multiple lines of defence before he/she is able to breach the confidentiality, availability, or integrity of the NATO, NATO member's or coalition partner CIS(s).

Generic IDS Model

The generic IDS model from [9] is visualised in figure 2. The purpose of this model is to function as a common viewpoint for the study and discussion of functionalities and components of IDS in a product independent way. Different components are distinguished and described in a logical order. Furthermore some additional definitions are provided.

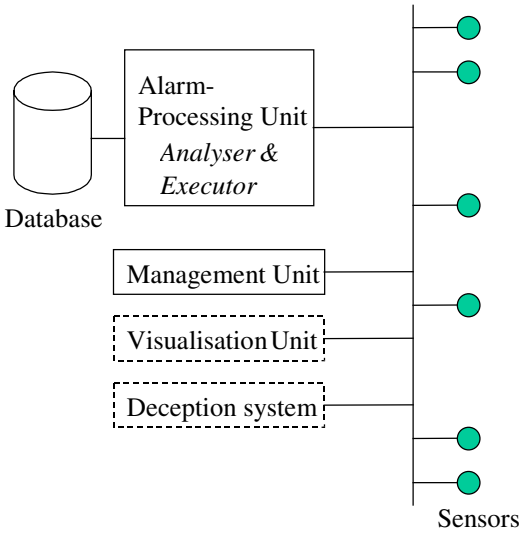


Figure 2: IDS Generic model

Sensors

The description of the generic components of an IDS is started with introducing *sensors.* These are the generic components of an IDS that collect *activity*. This activity can include network traffic, user misbehaviour, application misbehaviour and so forth. In the intrusion detection community it is common to distinguish between two types of activity: network activity and host activity.

Network activity: the activity present at the network is the network traffic, which can be categorised in:

- Low level protocols (ISO layers 2 though 4 e.g. TCP, UDP);
- Application and service level protocols (e.g. SMTP, HTTP, FTP);
- Content (of e.g. e-mail or web pages).

---

[1] Information forensics, reconstruction and recovery: the application of forensic techniques to investigate crimes involving, either directly or indirectly, information and communication technology (ICT).

[2] A workshop on Inforensics will be organised by NATO/RTO/IST/RTG-003 in October 2002.

Host activity: at the hosts (including clients, servers and routers) several forms of activity are present, caused by:
- Users: the person operating on a host, e.g. identified by a login account;
- Systems: hardware, operating system;
- Network services (e.g. PKI, DNS);
- Applications: e-mail, web browsers, and so forth.

Examples of sensors are network interfaces in promiscuous mode and tools that read log-files. An IDS can have multiple sensors. Based on the type of activity the sensors collect, the following classification of IDSs can be made:
1. *Host-based IDS (HIDS)*: the IDS looks at activity on a host;
2. *Network-based IDS (NIDS)*: the IDS looks at the network traffic either in (near) real-time or inspects via log-files at regular intervals;
3. *Hybrid* IDS: the IDS has sensors collecting host and network activity.

Alarm-processing unit

The alarm-processing unit is the generic component of an IDS that pre-processes and analyses the activity collected by the sensors. Furthermore, the alarm-processing unit controls the reaction to be taken by the IDS in reaction of a detected intrusion. The processes of analysis (by the analyser) and reaction (by the executor) are essential for an IDS and are described in more detail.

There exist two main classes of alarm-processing units based on the technique they use for analysing activity to detect intrusions [1] misuse and anomaly detection. These classes of IDS analyser techniques (misuse and anomaly detection) will be discussed below. But first, the difficulty of analysing activity with respect to false alarms is discussed.

No alarm-processing unit is infallible in analysing potential intrusion related activity. The alarm-processing unit in an analyser may fail to detect intrusions, or sound the alarm when no intrusion has occurred. Four cases in the operation of an analyser are distinguished (Table 1).

Table 1: Four cases in the operation of an event analysis.

	Intrusion	**No Intrusion**
IDS Alarm	An intrusion has occurred, and the IDS generated an alarm *(correct alarm)*	No intrusion has occurred, but the IDS has (erroneously) detected an intrusion *(false alarm)*
IDS Rejection	An intrusion has occurred, but the IDS has not generated an alarm *(false rejection)*	No intrusion has occurred and the IDS has not detected an intrusion *(correct rejection)*

Related to these four cases two parameters of an IDS are defined:
- The *accuracy* of an IDS: the number of *correct alarms* divided by the number of *correct alarms plus false alarms*; is a parameter for the relative number of correct alarms. The more accurate an IDS is, the fewer false alarms it generates and the higher this parameter is.
- The *completeness* of an IDS: the number of *correct alarms* divided by the number of *correct alarms plus false rejections*; is a parameter for the relative number of correct alarms. The more complete an IDS is, the fewer intrusions remain undetected and the higher this parameter is.

In the ideal case, an IDS would be 100% complete (it detects all intrusions) and 100% accurate (it produces no false alarms). However, detecting an intrusion is a very difficult task. This comes partly forth from the base-rate fallacy problem described in [4]. The base-rate fallacy problem shows the need for very accurate IDSs. If

an IDS generates too many false alerts, the operating and managing personnel will have no confidence in the system!

Analyser techniques

Analysers that use the *misuse detection* method operate by searching for very explicit activity and/or patterns of activity. Misuse detection is also called detection-by-appearance.

A number of known *intrusion patterns* (also known as intrusion *signatures* or *rules*), that specify the features, conditions, arrangements and interrelationships among activity that leads to break-in or other misuse are stored beforehand in the IDSs knowledge database. The IDS collects activity and looks whether one of the stored intrusion patterns occurs. If an intrusion pattern is detected, the IDS will generate an alarm.
A detection-by-appearance IDS can only detect *known* intrusions, but once it detects an intrusion, it can usually specify exactly how the intrusion has occurred.
In a sense the misuse detection concept is paradoxical, because the intrusions have to be known beforehand. One could argue that a CIS should not be vulnerable to known intrusions. However, in practice it can be infeasible to remove all vulnerabilities from a system, because this is costs to much resources. Therefore the misuse detection IDS can certainly provide an important role to detect known intrusions, intrusion attempts and other intrusion related activity.

An advantage of a misuse-detection IDS is that it is not only useful to detect intrusions, but that it will also detect intrusion attempts; a partial signature may indicate an intrusion attempt. Furthermore, the misuse-detection IDS could detect port-scans and other events that possibly precede an intrusion.

A disadvantage of a misuse-detection IDS is that only known intrusions are detected. No protection is offered against novel attacks, or new variants of existing intrusions. More crucially, a small variation in the form/structure of an attack can invalidate a signature.

Misuse detection is the most used technique in current NIDSs. There are two different types of NIDS, smart and raw, depending on whether they look at patterns in low-level protocol activity or application-level protocol activity [5].

Smart: IDSs that have logic implemented that understands the target protocol. They will parse the request and perform (optimised) signature matching based on known rules pertaining to the protocol. They will attempt to behave like a real web server would behave, at the expense of additional code and slowness.

Raw: Also referred to as 'packet grep' style IDSs, they typically just scan the unprocessed raw data for key strings. The benefit of this is speed only. The term raw is not used in a derogatory manner, but rather to identify that these IDSs usually deal with the raw data directly, rather than interpreting the protocols.

Event analysers that use the *anomaly detection* method operate by examining the behaviour of the activity. Anomaly detection is also known as detection-by-behaviour. The ‘normal’ behaviour pattern of activity is stored beforehand in the IDS. The current activity is then continuously collected in real-time and analysed to see whether its behaviour significantly deviates from the stored behaviour. The IDS signals significant deviation between these behaviours as a (possible) intrusion.

An anomaly detection IDS is confronted by two related problems:
1. *Description problem*: How to describe the behaviour of activity in an effective and efficient manner?
2. *Comparison problem*: Given a stored behaviour pattern of activity and a current behaviour, when do these two deviate enough to constitute a possible intrusion?

An IDS based on anomaly detection is classified according to how it deals with these two problems, see [7].
The main problem with anomaly-detection IDSs is that it is hard to describe the ‘normal’ behaviour of activity, because of e.g. the unpredictable behaviour of the end-users. This results in IDSs generating a lot of false alarms. This is the main reason that anomaly detection IDSs are hardly used in practice and are mostly at a research stage.
The main advantage of a good working (few false alarms) anomaly detection IDS would be that in a sense ‘unknown’ intrusions can be detected. The systems however detect only the fact *that* an intrusion has occurred

rather than *how* it has occurred. This creates a situation where the IDS does not need a priori knowledge of specific security flaws in a CIS.

In [4], a strict anomaly detection model is described which has the distinct feature that it generates no false alarms by definition. The proposed strict anomaly detection model is as follows. An IDS should use precise definitions of 'use' for activity in a CIS, in accordance with security policy. Any deviation of these definitions is a security policy violation. In case of the normal anomaly detection the 'use' behaviour is not strictly defined but merely a description of normal behaviour of activity. The key-advantage of the strict anomaly detection model is that new attacks can be detected, while no false alarms are generated.

Human interaction with the analyser can be an important aspect of analysing activity. Therefore the alarm-processing unit should be able to cope with human intervention in the decision process of whether activity indicates an intrusion.

A characteristic of an IDS is the frequency of the analysis. Three categories are distinguished:
1. *Continuously*: events are collected and thereafter analysed, as they occur, - often in real-time.
2. *Periodically*: events are collected and analysed periodically from the subject. An example is system log files that are analysed every hour.
3. *Initiated under special circumstances*: e.g. when the system administrator suspects an intrusion.

The information about known intrusions and/or the normal behaviour of the activity is stored in a *knowledge database*. The alarm-processing unit can also store (information about) collected activity in a *storage database* that can be of interest in the future for example in a digital forensics investigation.

Reaction

When the IDS decides that certain activity indicates an intrusion, an alarm is generated by the executor component. This alarm can either be *passive* or *active:*

Passive: an IDS generates an alarm, which can be a log file message, a pop-up screen, a pager message and so on, or a combination thereof.

Active: the generic model includes IDSs that have (optional) active components that can generate automatic reactive control signals. These control signals could for example tighten a BPD, increase the IDS' sensitivity, shutdown a connection, divert network traffic to a decoy system or shut down hosts that are under attack.

An IDS can operate in close co-operation with network management systems. Alarms can be incorporated in network management systems. Active IDS-alarm components might send network management control messages to different components of the network.
In general a reaction of an IDS will be closely related to (response) management of the system. Although a reaction can be partly automated, in practice human interaction and decision making will be needed to come to a balanced reaction.

Management

The management of IDSs is crucial for efficient and good deployment of IDSs in military (and also government or corporate) environment. The management of an IDS is divided in four categories [1]:
- Detection management
- Response management.
- Update management.
- Availability management.

Besides the issues of scalability and protection of management operations is described.

Detection management involves communicating with the IDSs via e.g. a graphical user interface that visualises possible intrusions. Furthermore it can involve manual analysis of data by a manager, e.g. to double-check IDS alarms.

The intrusion detection system, when it is signature or rule based, has to be updated very regularly. As we already noted, new attacks arise every moment, hence the updating of signature based IDSs is an ongoing task.

The process of updating the system is called update management. Commercial IDSs do not have a continuous 24h updating process of signatures.

Once an intrusion is confirmed, responses have to be managed. Most of the actions that have to be taken can not be performed automatically by the executor-part of an IDS, but require manual intervention

Availability management deals with ensuring that the system is available at all times. Both the hardware and software components can go out of service and then need maintenance. Furthermore, IDSs can be under a denial-of-service attack.

Central management is an important property for IDSs in a CIS. Especially when scalability of the intrusion detection capabilities are concerned. Maintenance and updating of different systems becomes difficult when network environments are growing and IDSs are not centrally managed. As there is no standard management protocol for IDSs, IDSs from different manufacturers can not yet be managed from a single central management system.
Management operations themselves must be protected from intruders as well. For example, if management commands travel over an in-band network, there is a risk that the IDSs themselves get compromised, this includes the vulnerability to denial-of-service attacks. This issue becomes especially important when IDSs are centrally managed. Authentication and confidentiality can be provided by using encryption. In the mean time, most commercial IDSs already have encrypted communication implemented.

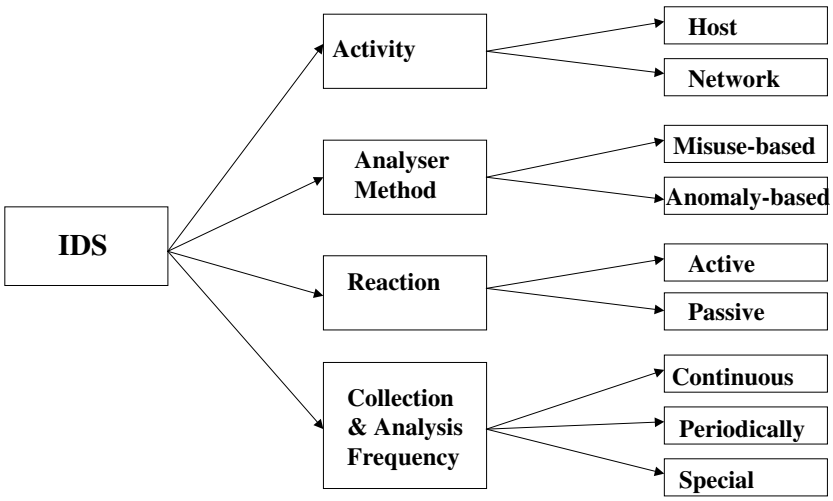


Figure 3: IDS characteristics

Visualisation
Closely related to the management of an IDS is the optional visualisation of intrusions. In order to obtain informational situation awareness, visualisation of intrusions in a CIS is an important issue. This is incorporated in the generic models for Intrusion Detection by the optional intrusion visualisation unit. It can combine information of intrusions with other information such as the network (link) performance (e.g. availability and usage). The visualisation component can be tightly linked to an IDS alarm processor, but preferably also able to fuse information from various types of sources, such as network performance monitors.

Deception systems
As an optional generic component an IDS can contain a deception system. These can be used to attract attackers to certain parts of the infrastructure. Preferably the deception system is on a stand-alone system. This has the advantage, that all network traffic directed to the system is suspicious and indicates an intrusion. In this way, an intrusion can be detected at an early stage. The entire intrusion can be recorded and hence novel intrusion techniques can be learned. Moreover, when the attacker spends time intruding the deception system, valuable time is gained over the attacker. This time can be used to protect the real CIS and/or to trace the intruder.

Deception systems, however, have disadvantages as well. Firstly, when compromised they can be used as a stepping stone to further compromise the CIS. Building the system in a virtual machine (jail) can make this a lot harder for the attacker. Secondly, deception systems add complexity to the CIS. This may lead to increased vulnerabilities. Finally, deception systems have to be managed at the cost of resources.

Distributed hierarchy of IDSs

Like the figure above of the IDS generic model, Figure 4 visualises a distributed CIS where each sensor consists of an IDS operating at a lower level of abstraction in the network. This distributed design is often seen in commercial products.
In a similar way, different alarm messages from a distributed IDS corresponding to different CISs can be communicated with an IDS one level of abstraction higher, by adding an extra layer to the model.
For instance, this could be a NATO-wide IDS where for instance different NATO member states send intrusion alarm messages to a NATO-wide intrusion visualisation and management facility. This allows management of intrusion detection at NATO level.

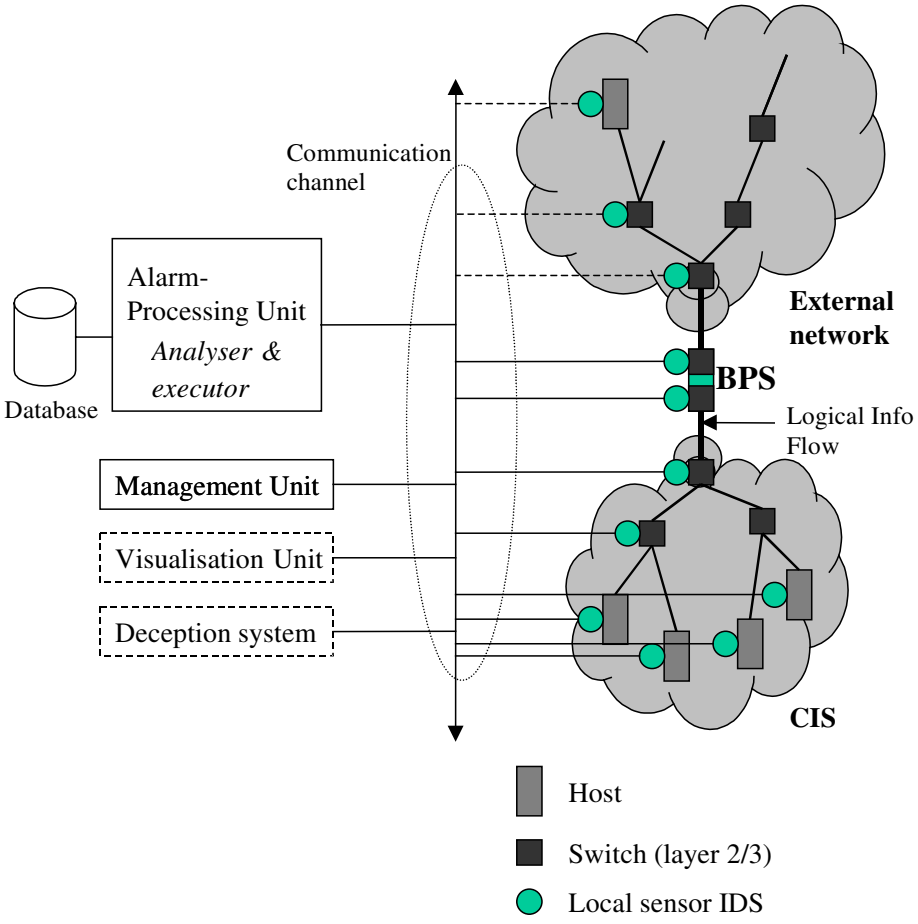


Figure 4: Distributed IDS

Another example is an IDS in a NATO military operation, where CISs of different NATO members, NNNs, NGIOs and NGOs release (parts of) their local intrusion information to the higher IDS-management and visual layer. This in order to acquire an overall operational awareness of enemy information warfare activities.

Real-time detection, a necessity

Before discussing the impact of the real-time necessity on IDSs, the Logicon Inc. time-axis model is used to describe the time-line of an attack (~ a deliberate intrusion) [6]. This attack scheme is presented in Figure 5.

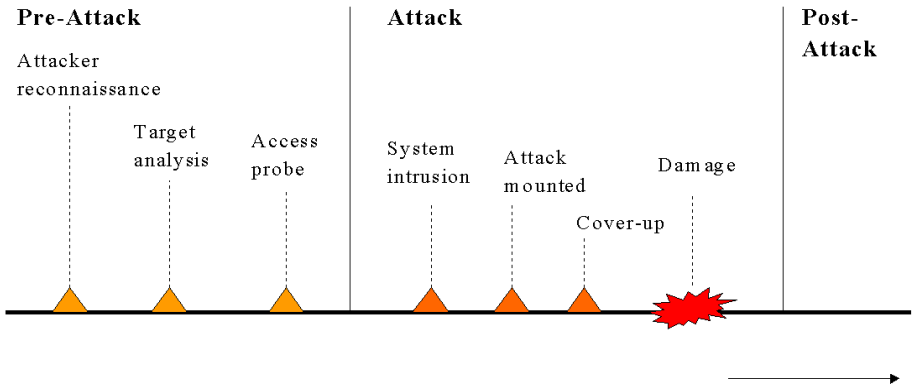


Figure 5: Time axis model of an attack (source: [6])

The actual attack is preceded by a pre-attack stage. The attacker will begin this stage by defining an end-state with regard to the CIS (~ target). This end-state is a clearly defined and obtainable objective. Desired results may be denial-of-service, acquisition of sensitive information, and/or establish and maintain access to the CIS.

After setting the objective the attacker will seek to identify and define problems associated with breaching the target defences, gather information and make assumptions about the CIS, develop possible courses of action (COA), and analyse each COA. In the time-axis model, three steps are distinguished in the pre-attack stage:
1. *attacker reconnaissance,*
2. *target analysis,*
3. *access probing.*

In the first step: *attacker reconnaissance*, the attacker starts acquiring critical information about the CIS. This includes execution of most, if not all, of the following steps: foot printing, scanning, enumeration, vulnerability mapping, and social engineering (i.e. using social skills to obtain info from e.g. employees). The second step: *target-analysis* consists of analysing the available information, making assumptions and then developing multiple COAs. In the third step: *access probing*, the attacker tests the COA, and then selects the best COA. The testing is often done, by sending probes to the CIS or by stimulating the CIS. For a “complex” attack, the pre-attack stage can last a long time. In the following stage of the time-axis model, the actual attack starts. Hereafter, the attacker will try to cover up the operation and/or leave a backdoor (e.g. a Trojan Horse or a kernel patch) in the system. An attack can be very hard to recognise when the cover up operation is performed well. After the damage is done, the post-attack stage starts. This is the stage, where the defender will try to take corrective measures and so forth.

The time-axis model can be related to the incident cycle described above. Both models have a point where an intrusion leads to damage. Where the incident-cycle models the defender’s actions in the periods of time before and after the damage, the time-axis model shows the attacker’s actions for these two periods of time. The point of detection of the incident from the incident cycle can also be related to the time-axis model. When an attack is detected in the pre-attack stage, this is called *pre-attack detection*. Similarly when an attack is detected in the actual attack stage this is called *attack-detection*. It is also important to recognise that a system was attacked and that possible damage has occurred or that there is a security breach. The detection of an attack in the post-attack stage is defined as *damage detection* or *post-attack detection*. Has crucial information been modified? Is there a backdoor present in the system?

It is crucial for defence organisations, but also for most other organisations, that intrusions and attacks are detected as fast as possible. It is impossible to take reaction, correction and evaluation measures if it is unknown that an intrusion will occur, is occurring or has been occurred. Hence, damage can not be controlled and minimised, which might result in financial disasters, or worse.

For intrusion detection this implies that IDSs should operate in a real-time manner. Real-time intrusion detection can be seen from two viewpoints. Firstly, within the boundaries of technology an alarm should be available to the response managers as soon as possible. And secondly, an intrusion should be detected as early as possible on the timeline of an attack. An important property of the analyser to achieve the latter is the ability to correlate data.

Correlation of data

At most stages of the attack, there is activity in the (target) CIS, which can be detected. The process of interpreting, combining and analysing the information of all available sources (such as IDS in the target CIS, but also available information from other sources) is called *correlation* or *fusion*.

Different sensor IDSs (located in the CIS or even in external network) can collect information from different stages of the attack. To optimally use this information for early warning the IDS should be able to correlate the information in real-time. This should especially include information from the pre-attack stage, since the first signs of an attack are visible in this stage. The information used could be *in-band information* or *all-band information* [8]. In-band information is all information from activity inherent to the target system. All-band information can be any other information that can be used in the correlation, including information from human intelligence sources.

Conclusion

The generic IDS model, the different aspects of IDS and all issues mentioned above, shall give the general audience enough background for understanding the issues addressed by the other practical and research papers later during this symposium. For a more extensive introduction to the generics of IDS, the reader is referred to [9].

References

1. 'ISO/IEC PDTR 15947, Information technology – Security techniques – IT intrusion detection framework', ISO/IEC JTC 1/SC 27 N2691.
2. The President's National Security telecommunications Advisory Committee, Network group Intrusion Detection, 'Subgroup Report on the NS/EP Implications of Intrusion Detection Research and Development', December 1997.
3. 'Technical and Implementation Directive for the Interconnection of Communication and Information Systems (CIS)', INFOSEC.
4. Sasha, Beetle, 'A strict anomaly detection model for IDS', Phrack 56, source: www.phrack.com., 6-11-2000.
5. Rain Forest Puppy, 'A look at Whisker's anti-IDS techniques', 1999, source: www.wiretrip.net/rfp/, 6-11-2000.
6. Paul Zavidniak, Logicon Inc.: 'Achieving Information Resiliency in the Defence Environment', Information and Security and Data Security Congress, February 2000.
7. Esmaili, M., R. Safavi-Naini and J. Pieprzyk, 'Intrusion detection: a survey'. In: S.J. Chung (ed.). Proceedings of the 12th International conference on Computer Communications 21-24 August 1995 in Seoul. Amsterdam, 1995, pp. 409-414.
8. Amoroso, 'Intrusion Detection: an introduction to Internet surveillance, correlation, traps, trace-back, and response', intrusion.net books, 1999.
9. Coolen, R., Luiijf, H.A.M. (2001), *Intrusion detection, Generics and State-of-the-Art*. NATO RTO/IST TR-49, NATO RTA, Paris, France. On-line: www.rto.nato.int.

Click here to view PowerPoint presentation; Press Esc to exit

Slovak View on Real Time Intrusion Detection

Maj. Ing. Pavel Šimon, PhD.
Military Technological Institute
ul. kpt. Nálepku
03101 Liptovský Mikuláš
Slovakia

mailto:Simon@mti.sk

Mgr. Eva Triznová
Military Technological Institute
ul. kpt. Nálepku
03101 Liptovský Mikuláš
Slovakia

mailto:Triznova@mti.sk

1. Summary

Increasing use of CCIS is creating both new capabilities and new vulnerabilities. All professionals from IT branches must accept this real situation. For the government and especially military organisation arises a new role in prevent and defend of information systems - their own systems and data in these systems. However many small military or other organisation have only small possibility to use commercial IDS.

Topic covered by this article: Real-time decision support; On-line visualisation of intrusion and progress thereof; Information fusion, including correlation, aggregation, and alignment techniques.

2. Aims and objectives

The basic aim of this article is: different point of view of Intrusion Detection System, Especially small independent part of military network's information technology manager's point of view.

3. Basic terms

Internet is very useful for study and increasing security, but it gives to attacker many tools to attack. In history attacker (hacker or cracker) was someone having big experiences in computer science especially in network and security (these are called "guru"). Today many attackers don't have many skills (in some case attacker is able only to use web browser and click on mouse). There is large number of attack tools on internet. Commonly quality was changed to quantity today. However, attacker "guru" is now better than anytime in the history.

Intrusions generally fall into two categories: misuse and anomalies. Misuse attacks exploit some vulnerability in the system hardware or software to gain unauthorized access. Many of these attacks are well documented and are easily detected by computer systems, but new ones are constantly being discovered. It is more difficult to detect anomalies since they more frequently originate from an inside user who already has access to the system. They are characterized by deviations from normal user behaviour, and detection requires some type of user profiling to establish a standard behaviour pattern.

It is more easily to detect misuse than anomalies, because misuse is type of attack, but anomalies is right using of system with non-standard behaviour. In common way misuse is from outside and anomalies is commonly from inside of systems.

For administrators of information systems is not important, whether the attack is intentional or not, because in both case is important to respond to intrusion.

There is a variety of different probes that hackers will attempt:

The first type we will prepare for is one of the most common, **port scans**. Port scans are where an individual attempts to connect to a variety of different ports. The scans can be used on a specific target, or used to scan entire IP ranges, often chosen randomly. This is one of the most

Paper presented at the RTO IST Symposium on "Real Time Intrusion Detection", held in Estoril, Portugal from 27-28 May 2002, and published in RTO-MP-101.

popular information gathering methods used by hackers today as it identifies what ports and services are open.

The second is **brute force attempts to login**.

Another example is the common **/cgi-bin/test-cgi** attack used on web servers.

Lately many attacks are **using** some **exploits** in network system.

Generally attacks can be divided into groups showed in Table 1.

One-to-one	Attacker uses a single machine to attack a single target machine. Example: sendmail bugs.
One-to-many	Attacker uses a single machine to attack many targets. Example: probes, denial of service attacks.
Many-to-one	Attacker divides assault among multiple outside machines to attack a single victim. This is difficult to detect because multiple connections from multiple sources look more innocent than multiple connections from a single source. Example: SYN flood using IP spoofing to deny services, DoS attack.
Many-to-many	Many collaborating attackers divide the tasks of probing/attacking multiple victims. This poses the same challenge as the "many-to-one" case with the added complexity of multiple target machines. This kind of attack is very difficult to detect. Example: "Smurf" attack from multiple sources.

Table 1: Attack Types (from [Durst, 1999])

There are three kinds of intrusion-detection products: host-based, network-based and distributed systems. Every another system can be considered as a combination of these three products.

Host-based products have an agent running on each protected host. The agent sends a regular heartbeat, as well as alarms, to a management station. The heartbeat ensures that the management station can detect a denial of service aimed at overwhelming a host so that it's unable to respond or work normally.

A host-based system resides on a single host computer. It uses audit logs or network traffic records of a single host for processing and analysis. This type of system is limited in scope since it is able to see its own host’s environment only, and cannot detect simultaneous attacks against multiple hosts.

Network-based monitors are sitting on the network capturing packets and trying to find match between current situation and known attack patterns. They generate alarms when they see something suspicious too and may also send a heartbeat to a central console.

A network-based system is a dedicated computer, or special hardware platform, with detection software installed. It is placed at a strategic point of a network (like a gateway or sub network) to analyze all network traffic on that particular segment. It can scan data traffic for known attack patterns. It can also determine Internet Protocol (IP) addresses that originate outside its subnet. This system can detect attacks against multiple hosts on a single subnet, but it usually cannot monitor multiple subnets simultaneously. It also cannot detect any host-based attack that does not pass through it.

Distributed systems allow detection software modules to be placed throughout the network with a central controller collecting and analyzing the data from all the modules. This provides a robust mechanism for detecting intrusions across several subnets and several hosts. But it requires a dedicated computer to act as the central controller; centralization can make it vulnerable to attack.

Other approach is to divide IDS into two categories by needed equipment – dedicated system or part of network architecture. **Dedicated system** is running as separate part of network with hardware support. **Part of network architecture** is running on some network unit, for example firewall, proxy server or web server. This paper is trying to describe only IDS as part of network architecture, not dedicated system.

4. Architecture IDS for small unit

Military W/LAN is based on a cellular technology (fig. 1.). This provides good condition for using network-based IDS.

The administrator of cell is responsible for security and integrity inside the cell. He (or she) has to follow security proposal in military W/LAN, but he has large area for increasing the security.

From military W/LAN point of view distributed system is the best type of system, but) the W/LAN has very complex, difficult and heterogeneous architecture. This is the reason for using multilevel network-based IDS. The highest level is at gateway W/LAN to another network. Lower level is at connector cells to W/LAN. These places are ideal for using network-based IDS.

5. Part of IDS

Studying and utilizing IDS we found out incongruousness in terminology. The understanding of the terms is not unified. The elements like Control Sums System are considered by some authors as a part of IDS, but other repudiate it. The similar situation is in individual point of functional IDS.

Intrusion Detection System has many parts, but for this article it was decided to define three important steps:

- Detect intrusion

  When intrusion-detection products detect something suspicious, they notify you, typically by pager, e-mail, or an SNMP trap. Intrusion-detection products may have automated responses also that can cut short a hacker's visit to your network within milliseconds.
- Visualisation of intrusion

  When intrusion is detected, the first thing is to give support for administrator’s reaction. This means: visualise type, level of importance and other important information.
- Decision support of action

  Previous steps provide a good condition for making the decision in choosing support tools. Administrator needs to know much information for the decision in every case of intrusion.

Each tool in next chapter will be described in above mentioned three steps. Second point of view to tools is whether the particular tool is usable in real-time or not.

6. Example of tools

In field of IDS exists huge number of commercial (or COTS) tools. The prices of these tools are different, but in general, they are very high. On the other hand there are many valuable tools with low price to disposal. These tools were designed and developed by internet community around Open Source community.

Most of tools described below were developed as Open Source, or distributed with GPL licence. Many of these are distributed as source code and administrator that has experience in programming is able to check their integrity and level of security. These features give administrator better feeling in security – no black hole, lower ability of exploits and many others possibilities. Security depends on administrator, not on people of some outer company. However from other side this requires more effort of administrator. He has to study fields of security.

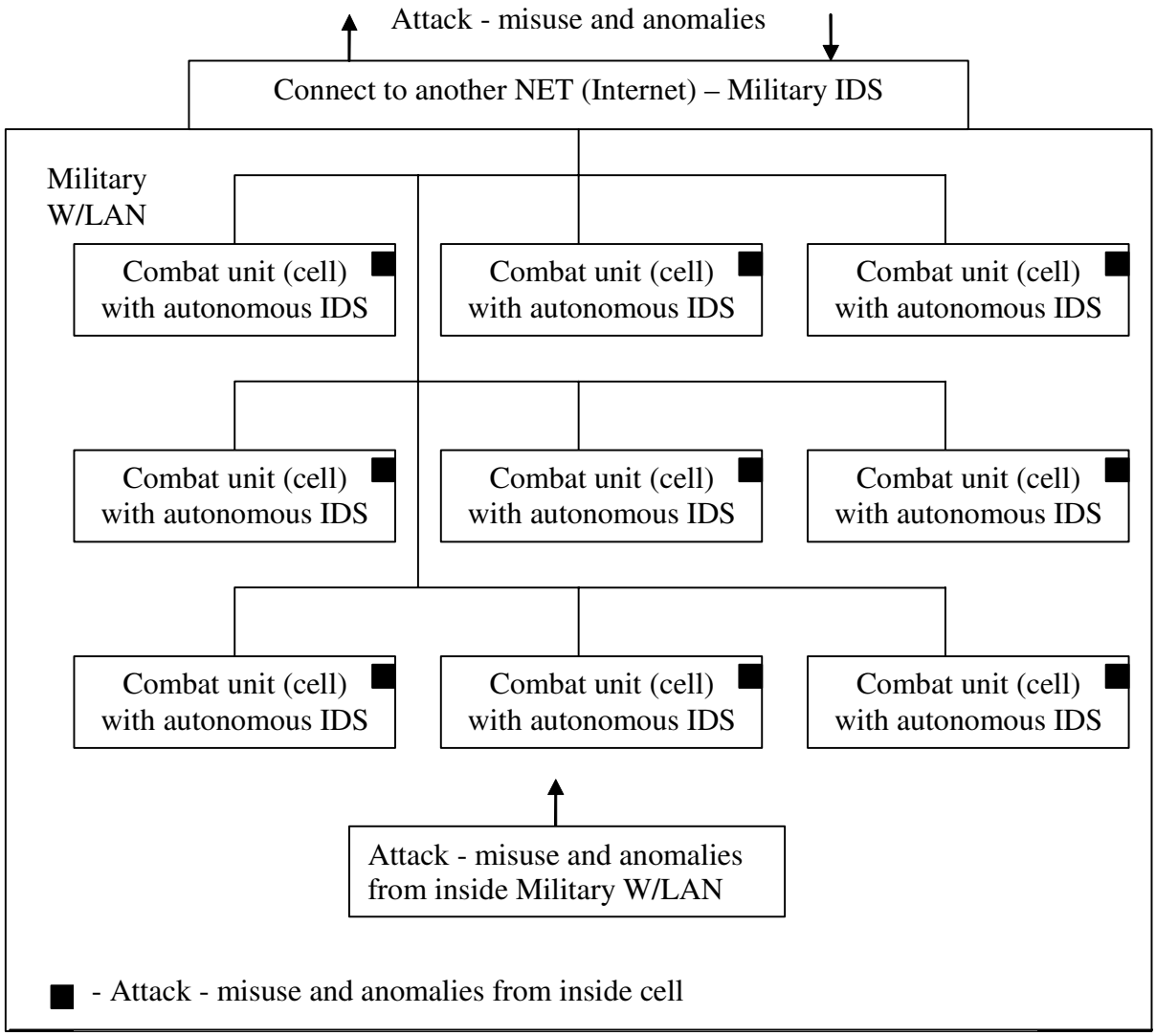


Fig. 1. Architecture IDS for small unit

First of all it is the easiest tool - **TCP Wrapper**. Created by Wietse Venema, TCP Wrappers allows us to control, log, and most importantly, react to any wrapped service. When someone connects to one of the services we defined, TCP Wrappers will log the connection (via syslog) and then spawn our alerting mechanism (for example via e-mail). This simple tool is participated in all three parts of above defined IDS, but the second (visualisation) is very simple. In addition it isn't real-time procedure because it works only via e-mail.

Next tool is **TEA – Third Eye of Administrator** (tea.sourceforge.net). It works using more simple mechanism than TCP Wrappers. Tea generates perl-script 'checker' based on config file, then Checker process input and performs required actions. Loosely said, config file consists of two types of rules: ***Base rule type and cumulative rule type***. Base rule type: This type of rule immediately performs action depending on input. Rule is pair (regexp, action). When input line pass through regular expression (next only selector), perform action. This kind of rules is useful only for ignoring carefully specified known events which appears frequently. Cumulative rule type: cumulates input lines until additional conditions (next only modifiers) are passed and then perform an action. Rule is triplet (selector, modifiers, action).

TEA works in both online and offline modes. Online mode offers real-time system monitoring. Input is activated base type rule and immediately invokes action defined in rule. When input activates rule of cumulative type internal structure (next only event-str) is created and input processing continues. Event-str contains information describing input (number of accepted inputs, timeout until last input, etc.). Identifier of event-str is concatenated from substrings of input line

binded by '()' in rule selector. Offline mode is also powerful tool, but isn't important for real-time IDS.

TEA gives good help to administrator in all of three part IDS. It is strong tool, but the biggest groin is visualisation.

Previous tools are based on scan system logs a technology, better IDS is based on promiscuity mode of network card.

One of active tools is **LSOF**. This tool woks only in one of the parts – detection. LSOF is very useful utility. It helps us in many different situations. It is able to find which files, peripherals or communications lines are active or which programs are in using in the time.

It is suitable to reconsider active processes from time to time. It is useful to explore whether suspicious TCP port haven't been opened by active processes, whether they haven't saved something in strange files, etc. LSOFT is the freeware and you can get it on site **ftp://vic.cc.purdue.edu/ /pub/tools/unix/lsof**.

Another network IDS package is the **Shadow** system. Shadow grew out of another project called the Cooperative Intrusion Detection Evaluation and Response (CIDER) project. While Snort (see below) provides for a real-time network intrusion detection capability, Shadow provides a near-real-time intrusion detection. This difference grows out of the fact that Shadow updates its information at a predefined interval (the Shadow documentation speaks in terms of one hour time intervals). On the other hand, Shadow provides for data analysis. This analysis helps reduce false alarms.

More complex tool is **SNORT.** Snort is defined as a "lightweight Intrusion Detection System". By definition, lightweight IDS should have a small system footprint, provide for cross-platform support, and easy installation. It utilizes the `libpcap` library (originally developed at Lawrence Berkeley Laboratory) for sniffing traffic and then analyzing the packet payloads. There are three primary subsystems to Snort:

1. Packet decoder,
2. Detection engine,
3. Logging and alerting system.

Good active system is **NMAP.**

Another principle is to make control sums, important sums and checking its integrity. Only make sums and check its integrity is non real-time, but with same additional warning tools it is possible to work real-time. The best known tool in this category is **Tripwire**.

AIDE stands for Advanced Intrusion Detection Environment, and the software is meant as a free replacement to Tripwire. AIDE originally appeared when Tripwire was a strictly commercial product. AIDE is written in C and provides a variety of features including using a Postgres SQL database as a back-end for storing the hash signatures of the files AIDE is configured to monitor. The AIDE tarfile can be obtained from the Web site: http://www.cs.tut.fi/~rammer/aide.html.

One of the best Open source projects in IDS is **LIDS – Linux Intrusion Detection System.** LIDS is a Linux kernel patch to enhance the Linux kernel. LIDS provides **Protection, Detection and Response** to the intrusion in the Linux kernel.

- **Protection.** LIDS can protect important files on your hard disk no matter what file system type they reside on, anybody including root can not change the files. LIDS can also protect the important processes from being killed. LIDS can prevent RAW IO operations from an unauthoritized program. It can also protect your hard DISK, include MBR protection, etc.
- **Detection.** When someone scans your host, LIDS can detect it and inform the administrator. LIDS can also notice any activity on the system which violates the rules.
- **Response.** When someone violates the rules, LIDS can log a detailed message about the violated action to the system log file which has been protected by LIDS. LIDS can also send

the log message to your mailbox. In this case, LIDS can also shutdown the user's session at once.

FCheck, written by Michael Gumienny, is another freely available file integrity checker similar to Tripwire and AIDE. FCheck has the capability to monitor files, directories, or even complete file-systems for any additions, deletions, or modifications. It is a Perl program and requires only one other file, the **`fcheck.cfg`** configuration file.

SWatch (short for the Simple Watcher) monitors the system log-files for any suspicious activity. SWatch is written in Perl and relies heavily on regular expression matching.

Logcheck, **PortSentry**, and **HostSentry** are products from Psionic Software. They are available from Psionic's Web site www.psionic.com. Each package is designed to perform a unique function. HostSentry reacts to login/logout activity by monitoring the **`utmp/wtmp`** files on UNIX systems, LogCheck spots problems and security violations by processing information in various system log files on UNIX systems, and PortSentry can detect and respond in real-time to port scans against a target host. Of the three packages, HostSentry is still in the very early alpha stage. Logcheck and PortSentry are mature packages that can be utilized across an organization's network.

7. Conclusion

This paper contributes other kind of possibilities to solve security of network. This article provides overview of problems with IDS. Most of the Slovak firms (defence including) don't be able to procure big and expensive commercial safety and security systems called "black box". However they have to be able to protect their systems in any way. We have found the good gold way between high requirements of security and relative cheap solution – Open Source solution.

In this case the Open source is not only synonym of low price. High quality and efficiency of these system's tools exceed commercial IDS. The disadvantage of the Open source solution is the necessity of very skilled administrator of network. However it is an apparent disadvantage, because an experienced administrator signifies a safer system.

This approach is at any rate totally different from commercial systems. They usually rely on external experts and service from special IT firms. There are cases that even the administrator of network doesn't be able to influent events which are running in the system.

References:

1. **CAPT Dennis J. Ingram, H Steven Kremer, and Neil C. Rowe:** Distributed Intrusion Detection for Computer Systems Using Communicating Agents, http://www.cs.nps.navy.mil/ /people/faculty/rowe/c4i00a.htm
2. **Michael Hurwicz:** Cracker Tracking: Tighter Security with Intrusion Detection, http://www.byte.com/art/9805/sec20/art1.htm, May 1999
3. **Ingram, Dennis J.:** Autonomous Agents for Distributed Intrusion Detection in a Multi-host Environment, http://www.cs.nps.navy.mil/people/faculty/rowe/ingramthesis.htm
4. **Joseph Barrus, Neil C. Rowe :** A Distributed Autonomous-Agent Network-Intrusion Detection and Response System, http://www.cs.nps.navy.mil/people/faculty/rowe/barruspap.html
5. **S. Cheung, R. Crawford, M. Dilger, J. Frank, J. Hoagland, K. Levitt, J. Rowe, S. Staniford-Chen, R. Yip, D. Zerkle.** 1999. "The Design of GrIDS: A Graph-Based Intrusion Detection System." *U.C. Davis Computer Science Department Technical Report CSE-99-2*, 1999. [http://seclab.cs.ucdavis.edu/arpa/grids/grids.pdf]
6. **R. Feiertag, L. Benzinger, S. Rho, S. Wu, K. Levitt, D. Peticolas, M. Heckman, S. Staniford-Chen, C. Zhang.** 1999. "Intrusion Detection Inter-component Adaptive Negotiation." *Proceedings of the RAID 99: Recent Advances in Intrusion Detection.* [http://seclab.cs.ucdavis.edu/papers/tis99.pdf]
7. **Y.F. Jou, F. Gong, S.F. Wu, H.Y. Chang, et al.**, "Design and Implementation of a Scalable Intrusion Detection System for the Protection of Network Infrastructure", DARPA Information Survivability Conference and Exposition (DISCEX 2000), *IEEE Computer Society Press*, January, 2000. http://shang.csc.ncsu.edu/papers/desimpidsystem.pdf
8. **Durst, Robert, Terrence Champion, Brian Witten, Eric Miller, and Luigi Spagnuolo**, "Testing and Evaluating Computer Intrusions Detection Systems", Communications of the ACM, July 1999, Vol 42, No. 7, 53-61.
9. **Dusan Kotora.** TEA – Third Eye of Administrator. Documentation of system. http://tea.sourceforge.net

Click here to view PowerPoint presentation; Press Esc to exit

Experiences with Network Intrusion Detection

R. Coolen, H.A.M. Luiijf
TNO Physics and Electronics Laboratory
P.O. Box 96864
2509 JG The Hague, The Netherlands

E-mail: *coolen@fel.tno.nl, luiijf@fel.tno.nl*

W.J.F. v. Geloven, E.A. Bakker
Defence Telematics Organisation
P.O. Box 104
3140 AC Maassluis, The Netherlands

E-mail: *wjf.v.geloven@mindef.nl, ea.bakker@mindef.nl*

Abstract

This paper describes our experience with several commercial Network Intrusion Detection Systems (NIDSs) deployed in a network connected to the Internet. Specific problems in the operation of NIDS are highlighted, and a number of solutions to identified problems will be presented. Finally, we shall present our view on the contribution of NIDS to the security posture of the network environment to be secured. Throughout the paper, the focus will be on the real-time aspects of incident detection in networks, and, to a lesser extent, to incident response.

Introduction

Numerous papers describe the proliferating threats to military communication and information systems (CIS), including those of NATO, non-governmental organisations, and coalitions. Other papers describe the increasing importance of networks and information systems in military operations, e.g. voice and video communication between and among soldiers, equipment and vehicles in the area of operations, and with the command and control centres.
The Internet Protocol (IP) protocol suite is likely to be the future standard for military communication networks. The vulnerabilities of this protocol suite have been the subject of many research projects. The open nature of the Internet protocols with its inherent insecurity, its lack of authentication, the vulnerability for denial of service, and the like, is well known [1].
This paper describes practical experiences with network intrusion detection systems (NIDS). NIDS are means to detect, and alarm on, incidents occurring in a CIS. Although the experiences described in this paper result from the deployment of NIDSs in a static infrastructure, the results and lessons learned are equally relevant and applicable for (future) tactical IP-based networks. Throughout this paper the focus will be on the real-time aspects of incident detection in networks and to a lesser extent to incident response.

Description of the test network

The Netherlands Defence organisation has a nation-wide network, with approximately 3000 kilometres of glass-fibre, connecting Army, Navy, and Air Force, the Dutch Ministry of Defence, and other parts of the defence organisation. This unclassified, well-protected defence network is called the Netherlands Armed Forces Integrated Network (NAFIN-network, or NAFNET). The NAFIN-network is managed by the Defence Telematics Organisation (DTO).

To compare various Network Intrusion Detection Systems (NIDS), the authors received permission to connect a test network to the front-end router of NAFNET. The routing was set up in such a way that all traffic coming into the access router which is situated in front of the boundary protection device (BPD) was duplicated (mirrored) to the test network (see figure 1). All NIDSs under test were connected to this test network. As this network was located in front of the boundary-protection services, all technically erroneous packets and packets of attack attempts from and to the Internet appeared on the ‘test network’. In the perfect case, all NIDSs should see the same error and attack packets and should signal the same incidents.

Paper presented at the RTO IST Symposium on “Real Time Intrusion Detection”, held in Estoril, Portugal from 27-28 May 2002, and published in RTO-MP-101.

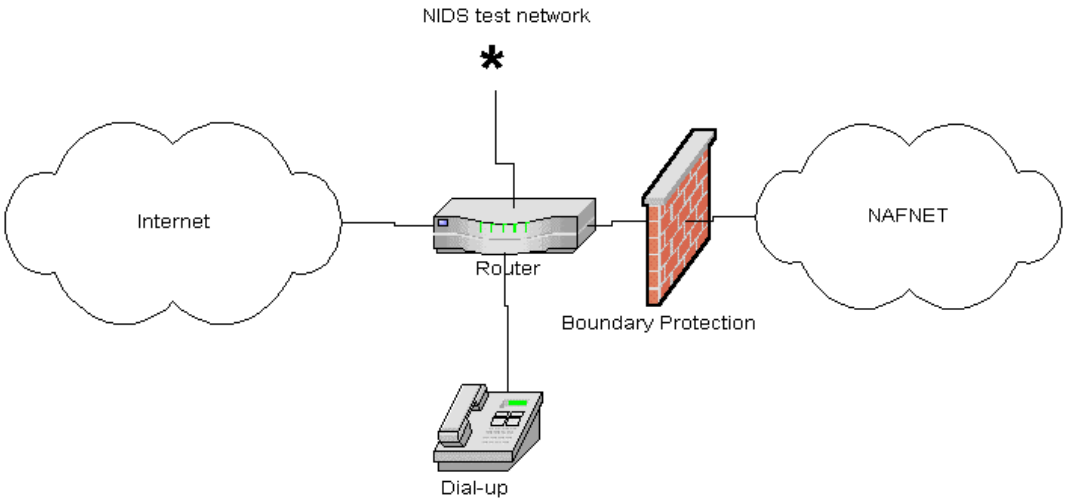


Figure 1: The NAFNET-Internet interconnection with the NIDS-test network

The experiences described below are from operating five Network Intrusion Detection Systems parallel and simultaneously. The experiments were executed in the spring of 2001. Below, we shall discuss our experiences in a non-product specific fashion.
Four of the NIDS products were commercial-off-the-shelf: Realsecure from ISS, BlackICE from Networkice, Netprowler from Symantec, Netranger from Cisco. The fifth product was an open-source NIDS called Snort [2].

The detailed layout of the test-network is depicted in figure 2 below.

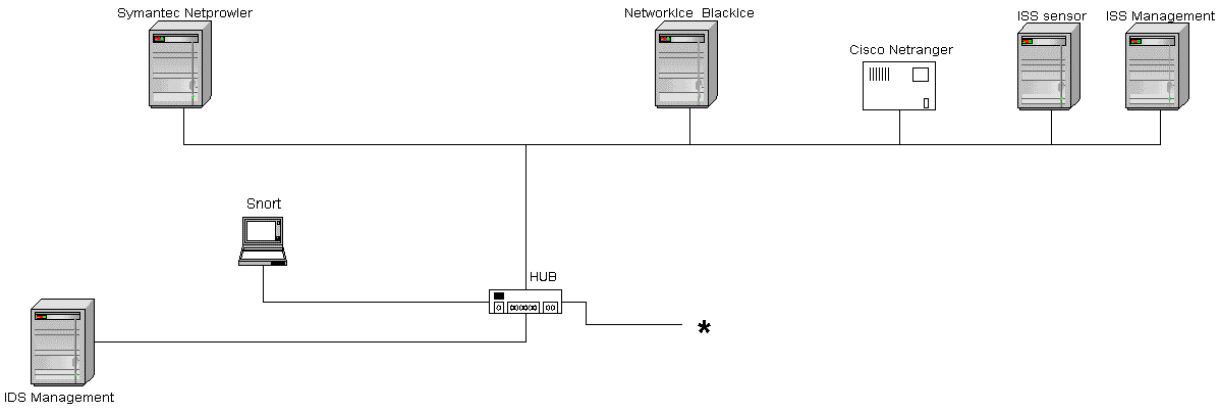


Figure 2: The test network

Testing network intrusion detection systems in a controlled laboratory environment involves the simulation of attacks and intrusion scenarios. We were however interested in real-life experience. Hence, the results of our experiments were dependent on the actual occurrence of erroneous and malicious packets which was, of necessity, beyond our control. In such a situation, it is difficult, if not impossible, to measure the accuracy (number of correct alarms divided by the total number of alarms) and completeness (of the number of detected incidents divided by the number of incidents that really occurred) of a product due to the lack of a controlled environment. In general, it is difficult to evaluate these aspects even in controlled laboratory environments as it is difficult to simulate sophisticated attacks. Furthermore, it is equally difficult to simulate realistic background traffic and a realistic background traffic load for the NIDS under test. MIT Lincoln Labs [3] evaluated NIDSs in a controlled test environment and in fact corroborated the above-mentioned difficulties with the simulation of attacks and background traffic.

The products used in our experiments are all signature-based. This means that they look for pre-registered patterns that could indicate a possible incident in the network traffic. Before a NIDS can detect an anomaly and signal an incident, it needs to know which byte pattern in (subsequent) packet(s) to look for in the traffic.

Such a pre-defined pattern is called a signature. Manufacturers of NIDSs usually deliver a few hundred known signatures together with the NIDS-products.

For an operational test of multiple NIDSs such as the one we envisaged, it was important to understand the overlap and differences between signature sets of these NIDSs, in order to understand differences in detected and undetected anomalies. To give an example, if only one of the NIDSs gave an alarm, this could mean that:
- the other NIDSs failed to detect the incident, or
- the alarm of the single NIDS that signalled an alarm turned out to be false,
- the other NIDSs did not have a signature of the incident.

The tested NIDS products typically consist of a management-console and one or more sensor-systems [5]. The sensor monitors the network-traffic for potential incidents. The management-console is used for the configuration of the sensors and for other NIDS-management tasks. The management display can be used for the real-time display of alarms as well, usually including an alert priority. Real-time in this case means: prompt detection of a possible incident by one of the sensors, communication of the alarm to the management-console, and then presentation of the alarm to the staff responsible for incident management.

We chose to let each product look at all signatures that were available from the vendor. Privacy sensitive signatures however were disabled. This approach has its advantages and disadvantages. The disadvantage was the increased complexity of comparing the incidents reported by the various NIDSs under test. The advantage of this approach was that we could learn more of what was actually happening on the network, and the way in which the various NIDSs behaved under identical conditions.

Problems with operating NIDSs

After the installation of the six NIDSs in the test network, we noticed that the NIDSs triggered a large number of alarms. Please note that this did not necessarily mean that we suffered from a range of incidents, or that damage was done. The alarms could be false; real incidents (including attacks) could have been stopped by preventive measures such as the boundary protection device (firewall), before reaching a target, and so on. These aspects will be discussed in more depth later in this section.

We observed that the large number of triggered alarms had a significant impact on the performance of the NIDSs. After a short period of operation, some of the products' databases suffered storage problems due to the huge amount of alarms. This required, amongst others, cleaning up the systems and making back-ups of the alarm data.
In earlier controlled laboratory experiments, we had already found that the NIDSs under test are very vulnerable to denial-of-service attacks. In our laboratory, we used the program Stick [4] to simulate thousands of attacks per minute. As a result, the performance of the NIDSs degraded significantly, in most cases resulting in a denial of service. The current experiments confirmed these problems in a operational test.

An important, initial observation during the experiments was that the NIDS products often disagree amongst themselves whether or not there was a possible incident and what type of incident that it could have been (even after taken into account the differences in signature sets).
Noticing this, we continued the experiments with a healthy suspicion about the accuracy and completeness of the NIDSs. A simple analysis showed us that, at best, one of the systems was accurate and complete. More likely, however, is that none of the NIDSs under test was accurate and complete. In order to solve the accuracy problem, i.e. to determine whether an alarm was false or not and whether the NIDS classified it correctly, we had to perform a thorough analysis of each alarm.

An alarm typically shows the following additional information on the management console:
- the name and/or type of incident
- source IP-address and protocol port
- destination IP-address and protocol port
- start time & date of the possible incident
- time & date of the last alarm of the incident sequence
- sensor that detected the incident
- priority

This basic alarm information is by far not enough to determine what really happened, whether security measures were breached, whether the target system suffered damage and who was responsible for the (potential) incident.

To accurately determine what caused the NIDS to trigger an alarm, additional information is needed about:
- The packet data that was analysed by the system and on which it based its decision to generate an alarm.
- The data that was *not* analysed by the system (missing some of the information, a NIDS could see that an intruder gathered a *PASSWORD* while in reality, someone just asked to *PASS the sWORD*).
- The signature and decision process of the NIDS: when and how does the NIDS consider something to be eligible for triggering an alarm? The signature is usually available to the user and often configurable, however the decision process and technical analysis details internal to the NIDS are unknown to the user. Although it is understandable from a business perspective that a manufacturer wants to keep these details company confidential, the use of, and the lack of access to, the NIDS can be limited by this absence. Note that since Snort is an open-source NIDS it does have the internal details available.

The following example illustrates the need for additional information with an alarm:

> *One of the NIDS showed a very large amount 'Stacheldraht' alarms. The NIDS could tell that 'Stacheldrah't is a distributed denial-of-service tool (DDoS). Further, desk research showed us that 'Stacheldraht' is a DDoS-tool that attacks systems with floods of ICMP traffic. Since we were quite certain that our test network was not suffering that many DDoS-attacks, we further analysed the traffic using a Sniffer application. This showed that a system in the network of the ISP was misconfigured, sending packets that contained the text-string: "This is not an attack". Still it was unclear to us what made the NIDS to decide that a DDoS-attack was occurring. In order to know this, a detailed study of the signature and the internal NIDS decision process would be necessary.*

Most NIDSs store some form of additional information with each alarm including the data bytes that the alarm decision was based upon. In general this is not enough to determine afterwards what had actually happened. For example, the data bytes that are not used in matching the signature, are often not displayed and logged. In some NIDS products the logged extra information is difficult to extract and to make visible, let alone to make it easily accessible through the user-interface. This implies that additional packet logging tools are required, such as sniffer applications. This also implies that, for proper incident management, a high level of expertise of both NIDS and protocol basics is required.

After having determined which packets were traversing the network and which ones triggered the NIDS, and whether the alarm was false or accurate, we still can not determine the potential impact of the possibly detected incident. To give an example, it is now difficult, if not impossible, to determine a priority for the response to an alarm. Even when an alarm is accurate, it is still unclear whether the 'dangerous' network traffic is able to pass or breach security measures between the location of the NIDS-sensor and the destination of the network-traffic.
This raises the question whether there is a relationship between the quality, capacity and potential interdependence of, and interaction between, the NIDS and the BPD. This important question needs further research as this combination is an implementation of the NATO security principle of defence-in-depth.

A question that needs to be posed, is: *can any damage occur?* To answer that question, detailed information is required about all of the following: the destination systems, their operating systems (versions and patch levels), the applications running on the various systems, all the configurations, and, last but not least, the security measures in place. Furthermore, the same information of the intermediate systems needs to be available to the NIDS-operators. To exemplify the foregoing, an attack on Windows NT will usually not harm a Linux-based system and an attack on an internal web-server will not reach its destination if a firewall filters the attack traffic.

After we analysed the accuracy of the alarm manually and assessed the potential damage of the incident, it is possible to determine (possibly some form of priority on) whether or not this specific alarm needs follow-up actions, i.e. incident response.
From this perspective, the real-time response aspect of NIDS has suffered a serious delay. Varying from minutes, if all necessary information is available to the NIDS operators, to hours if this information needs to

be gathered. Note that this is the case ***for every alarm!*** When a response to an incident needs to be initiated, this will typically include damage assessment, restoration and perpetrator/ attacker identification, digital forensics (also information forensics, or inforensics) and gathering of intelligence. These topics however are beyond the scope of this paper.

Determining the source of an attack can also be a problem. An attacker can use intermediate systems to hide its actual whereabouts, use falsified source addresses, and so forth. An incident response team can theoretically trace and identify internal perpetrators more easily, because the necessary information about users, network architecture, hardware addresses and the like is available to the organisation. The necessity of the latter is illustrated in the following example. We experienced that one system showed suspicious behaviour, in particular denial-of-service attacks, quite often. This turned out to be the proxy server, which by its nature, has to set up lots of connections for a large amount of users. This also means that if one of the users is involved in an incident, the proxy server accounting log needs to be analysed to find out the user's identity.

A small 'problem', after having analysed and responded to a single alarm, is that it would be useful to have some way of tracking alarms that have been already resolved. The current products do not have means to do this. This shows that most NIDS manufacturers do not have a full understanding for operational procedures, environments and settings of their products in a large infrastructure with heavy traffic.

The conclusion is that an alarm often needs to be analysed in detail. This process is often time, resource, and knowledge intensive. Furthermore, when dealing with 'too many' alarms, it will be necessary to have some form of prioritisation. These will be difficult to determine (and to define as well), because it will typically require information about the perpetrator(s), destination and intermediate systems, and the like. This information is often difficult to derive from the intruding log data that is supplied by the NIDSs.

Other problems with NIDS

A problem we address here is the scope or completeness of the NIDS-products. By design, the NIDS products search for patterns (or signatures) of (known) incidents in the network traffic. The use of heuristics, like anti-virus scanners do nowadays, is still in its infancy in NIDS-products. Hence, a prerequisite for detecting an attack is the availability of a proper signature for the type of attack. Consider a network with a wide variety of systems (different operating systems, hardware, and so on). Thousands of vulnerabilities have been published on the Internet (e.g. "bugtraq", "securityfocus"). A NIDS will generally only search for a few hundred of these attacks, often including signatures that are irrelevant for the specific network. As a consequence, the NIDS is far from complete in detecting possible incidents. In particular, novel or sophisticated attacks are not detected. Furthermore, the NIDS-products are lacking in detection of attacks for which they do have a signature. They are not 100% accurate!

When a new attack is published, a new attack signature is required to be able to detect the new attack with the NIDS. For obvious reasons, this signature has to be available to the system as soon as possible. Manual development of signatures is often possible. However, this would require a team of experts keeping track of new attacks and writing signatures. More convenient (although perhaps less reliable) would be that manufacturers update the signatures on a 24-hour basis. Currently, the update period varies from weeks to even some months! Can you imagine a virus scanner not being updated for weeks, let alone months?! Wouldn't the same hold true for NIDS-products?

It is disappointing to see that NIDS-manufacturers fail to update their products more frequently, devaluating the NIDS accordingly. Additionally, the manufacturers should allow users to customise their NIDSs to their own specific network environments, applications and operating systems, and hence bugs. Currently, this is not possible. The NIDS are far from complete in detecting attacks. In a way, this is extra worrisome as, by far, not all intruders and attackers are detected by the NIDS, despite the raised expectations.

A final problem we would like to mention here is the inability of NIDS to cope with encrypted traffic on the network with a NIDS-sensor. The use of encryption is increasing, with large manufacturers of software and hardware making the use of VPN (IPsec) connections easier to configure and use. In our specific situation this

means that the NIDSs were unable to analyse tunnels that were set up from the internal network to destinations throughout the Internet.

The aforementioned problems are relevant for all of the signature-based NIDSs involved in the experiments. Finally, larger differences among the NIDS-products can be found in the usability and clarity of the user interface and the co-operation with other security measures.

Solutions to identified problems

Having identified such major problems, one starts to wonder if there are solutions to these problems. And, if so, what are these solutions? Some of the problems discussed are a direct consequence of aspects of the experiments, and/ or can be (partly) resolved, as we shall discuss in this section.

Firstly, there are two problems related to the large number of alarms the NIDS products had to deal with:
1. Performance of the systems,
2. Amount of required resources for analysis (and response) of an alarm.

The obvious solution to these problems is to decrease the number of alarms generated by the NIDS, in particular the number of false alarms. A good way to reduce the alarms is to install more preventive measures in front of the NIDS, or vice versa, locate the NIDS behind preventive security measures, such as the boundary protection services (e.g. firewalls). This in fact provides a trade-off between the number of generated alarms on the NIDS and the number of detected ‘knocks on the outside of the network’. The priority and need to investigate incidents that only occur at the outside of the boundary of the corporate network can be low in case effective boundary protection measures block propagation of the incident to the corporate network. Locating the NIDS sensor behind the boundary protection to double-check the effect of the measure is nevertheless recommended in that case, as it forms an extra layer of defence. For example if a firewall blocks telnet traffic, the NIDS could have a signature that triggers an alarm if telnet traffic is passing the network (firewalls can fail as any other system), or it could be defeated and bypassed, in which case the NIDS should detect it.

Another way to reduce the number of alarms is to configure the priorities or signatures of the NIDS. It is essential for an accurate operation that a NIDS is fine-tuned to the specific environment in which it is deployed. Irrelevant signatures should be turned off, and tailor-made signatures may be required for specific mainframes or applications. Generally speaking, this requires a detailed threat assessment and vulnerability analysis of the ICT-infrastructure of the organisation deploying the NIDS. For example, the most relevant signatures can be chosen based on knowledge of the vulnerabilities that are present in the internal network. These signatures can then be configured with a high priority and alarms can be handled in real-time, whereas the low priority alarms could be aggregated in periodic reports. Determining priorities for attacks is, however, a non-trivial problem. Another way to fine-tune the NIDS is to configure thresholds for some signatures, which is possible for some types of signatures and NIDSs. A proxy-server for example has to set up a large number of connections and the threshold for a denial of service signature should be increased for these type of systems.

An obvious solution to the performance problem is to increase the performance of the systems. Of course, this solution is limited, but it may be cost effective. Equally, less human resources should be required to back-up or restore the system and so on. Note that whatever is done to reduce alarms, a significant amount of staff is still required to operate the NIDS (24hours x 7days a week). Furthermore, this staff needs a fair amount of technical expertise and knowledge of security. The problem of the NIDS vulnerability to denial-of-service attacks described earlier can be slightly limited by increasing the performance of the systems. However, the problem can not be resolved completely. It will always be one of the major disadvantages of signature-based NIDSs. It has also an important consequence: NIDS-systems shall be installed on systems separate of systems that offer application and other services. Otherwise, the NIDS can then be used as an instrument to attack those other applications and services. And, by the same token, the applications and services could be used in an attack on the NIDS. As a consequence, NIDS shall not be integrated in routers or installed on web servers.

A second area of problems, is the incompleteness and questionable accuracy of the products. The incompleteness stems from a more theoretical approach at signature detection. The accuracy problem was an

inescapable conclusion from the disagreement and discrepancy between the NIDS products. The solution to the accuracy problem is that which was discussed in depth when the problems with operating a NIDS were addressed. To determine whether an alarm was accurately triggered or not, the raw data, signatures, decision process of the NIDS, and preventive measures on intermediate systems have to be analysed. This requires good tools, methods, and information at the hands of the expert NIDS operating staff.

The lack of completeness has to be taken for granted in part, so that at least a false sense of security is not present. Manufacturers can be stimulated to update the NIDS signature database more frequently and more customisable. Creating signatures manually would require an expert-team. However, the problem can not be solved completely. The products cannot detect new, unpublished, or sophisticated attacks, but frequent updates minimise this shortcoming.

It is a positive attribute for a NIDS-sensor to be a passive box, i.e. it can only receive data. The upstream link to the network is preferably unavailable, e.g. by means of traffic filtering. Or the sensor can operate on OSI-level 2, thereby making it invisible to Internet-users. In the latter case, the sensors should be connected to the management-consoles out-of-band (preferably using another network than the network being monitored). The communications between management-console and sensor should be encrypted to ensure authentication, integrity (including non-repudiation) and confidentiality of the data. All these measures minimise the possibility that the NIDS itself is attacked!

A view on the contribution of NIDS

Despite the numerous, partly unsolvable problems, we still see value in installing a NIDS. Our general philosophy is that either the user benefits from the NIDS, knowing its limitations, or (s)he sees nothing or little of what is happening on the own network!

The first contribution to the security of the network which we recognise is the basic operation of the NIDS, i.e. searching for malicious patterns in network traffic at any location in the ICT-infrastructure. The organisation that provided the test network experienced more sophisticated attacks that triggered the NIDS. Although, the system classified the attack incorrectly and only detected a small portion of it, the alarm was still extremely useful. It told us that something ‘strange’ was happening on the network and the systems involved. Manual analysis led to the discovery of a larger underlying serious problem (which is beyond the scope of this paper).

A second specific contribution of NIDS is that it can be an extra layer of defence or an addition to preventive measures, such as firewalls and security-gateways. A NIDS can be tuned to detect failures or circumvention of preventive measures, for example by implementing the firewall rule-set in NIDS signatures. When a NIDS-sensor is logically located in front of, and behind, a preventive measure, it is also possible to execute a differential analysis.

A third contribution is that periodic aggregated reports of the NIDS can be used to show statistical information about amount of, and trends in, anomalies on the infrastructure. Although possibly relevant for decision-makers and operators (depending on the level of aggregation), this is obviously not what was primarily in mind when acquiring, installing and operating a NIDS for real-time intrusion detection.

Recommendations for improvement

Based on the experiences with the different NIDSs, manufacturers and developers of Network Intrusion Detection Systems should:

- Improve the performance and effectiveness of the systems. Networks are becoming increasingly faster and the throughput will increase accordingly. It is expected that legitimate traffic will increase relative to malicious traffic (it goes without saying that both will grow rapidly). This begs for better mechanisms of detection and faster systems. This point needs to be emphasised given the current large numbers of false alarms that the systems already produce.
- Make the details of the signature, the systems’ decision process, and the information that the alarm was based on, easily accessible through the user-interface. Furthermore, this information should be detailed enough to be able to quickly analyse whether an alarm is false. After all, time is of the essence, and is increasingly in shorter supply.

- Add a feature to keep track of the status of response to the system. As a minimum, it should be possible to provide an overview of the processed and unprocessed alarms.
- Perhaps most importantly, update the signature sets of the products more often and with extensive options for customisation of specific customer infrastructures.

Conclusions

Operating a NIDS, possibly with distributed sensors, in a large ICT infrastructure comes with major challenges. Some problems are situation specific, while others are relevant to all NIDS architectures and implementations in general.

Due to the size of the infrastructure, and perhaps its particular functionality, a 'large' amount of alarms needs to be dealt with. Partly, these alarms are false because the NIDS-products are inaccurate. Alarms may have a low priority, because they yield no damage or since preventive measures are located between the NIDS and target-systems. Additionally, not all incidents are detected, because the NIDS-products are incomplete. New and sophisticated attacks are not detected. The same occurs for incidents that are 'rare', or just have not been implemented in signatures by the manufacturers. Either way, a extensive amount of human resources, tools, methods, and knowledge is required to operate a NIDS. Hence, implementing and operating a NIDS is an expensive security measure.

Another set of problems is the performance issue. These issues are also related to the amount of alarms and traffic that needs to be monitored, as well as the vulnerability to denial-of-service attacks. Some of these problems can be partially resolved. The number of (false) alarms can be reduced by locating the NIDS behind preventive measures, and/ or by a good custom configuration and fine-tuning. Having done so, NIDS can have the following contributions, despite being expensive in both resources and knowledge:

1. *Basic intrusion detection*: either the user benefits from the NIDS knowing its limitations, or (s)he sees nothing or little of what is actually happening on the network!
2. *An extra layer of defence or an addition to preventive measures*: a NIDS can be tuned to detect failures or circumvention of preventive measures. When a NIDS-sensor is located in front of, and after, a preventive measure, it is also possible to see what attacks, that are detected by the NIDS before the measure, are stopped by this measure.
3. *Periodic aggregated reports by the NIDS*: these can be used to show statistical information about the amount of, and trends in, incidents on the infrastructure.

Acknowledgements

The authors would like to thank Peter Hupkens for his review of the paper and his valuable suggestions for improvement.

References

[1] Steven M. Bellovin (1989), *Security Problems in the TCP/IP Protocol Suite*. On-line: http://www.ja.net/CERT/Bellovin/TCP-IP_Security_Problems.html

[2] Snort web-site; www.snort.org

[3] Lippmann et al (2000), *The 1998 DARPA Off-line Intrusion Detection Evaluation*. RAID 2000 Proceedings, Lecture Notes in Computer Science, Springer, 2000.

[4] Stick; http://www.eurocompton.net/stick/

[5] Coolen, R., Luiijf, H.A.M. (2001), *Intrusion detection, Generics and State-of-the-Art*. NATO RTO TR-49, NATO RTA, Paris, France. On-line: www.rto.nato.int.

Click here to view slide presentation presented in PDF format

Embedding Policy-Controlled ID Sensors within Host Operating System Security Enforcement Components for Real Time Monitoring

Stephen D. Wolthusen
Security Technology Department
Fraunhofer IGD
Rundeturmstr. 6
Darmstadt 64283
Germany

Summary

This paper describes some attack and intrusion detection elements of a security architecture for distributed heterogeneous systems. The architecture concentrates on the level of the operating systems of the nodes involved and can also be retrofitted to existing COTS systems through the use of modular instrumentation extensions to the kernel and possibly the use of trusted coprocessor subsystems. The instrumentation provides both a reference monitor mechanism for active enforcement of security policies as well as sensor information for intrusion detection aspects, both of which occur under the control of a set of policies consistently enforced throughout distributed systems using external repositories. The reference monitor and intrusion detection mechanisms are controlled by policies defined in a first order theory permitting the abstract specification of subject, objects, and operations which are mapped to a given environment through the use of interpretations. This ensures a consistent enforcement of all applicable policies and permits the derivation of (consistent) additional rules based on automated deduction and can not only be used to model rule-based detection mechanisms but also to modulate the sensor output provided by the instrumentation within nodes. As an additional benefit, the use of predicates within the first order theory also permits a consistent view on observations at the time of data fusion.

Introduction

Since the proposition of automated intrusion detection [1] considerable research has focused on analytical mechanisms for the extraction of intrusion events [10,7,21] while particularly in host-based IDS the audit trail generation mechanisms have been presumed as largely immutable regardless of their suitability to intrusion detection. However, related research appears to imply that considerable benefits can be derived from using alternative sources of data streams for analytical steps such as the direct use of system calls [8,17,11.32] and the instrumentation of kernel components to record entry point transitions among these components [22] to the point where similar attractive results can be obtained even without the use of elaborate analytical techniques. The obvious disadvantages incurred from such techniques — besides the problem of embedding the instrumentation itself — are the large volumes of data one is confronted with for subsequent processing and the lack of inclusion of purely application-based observations that do not reflect adequately in operating system resource usage. However, they are more than balanced by the quality of the sensor data and their suitability to attack and intrusion detection provided that mechanisms for distributed processing of the sensor information are used.

The need for extensive instrumentation at the host level also derives from a changed environment in which systems must operate [2,33]. This is particularly the case in networking. Network IDS sensors are confronted with increasing reliance on switched components, causing a considerable increase in the number of sensors required. Also, new ad hoc networking mechanisms based on various techniques such as IEEE 802.11b, Bluetooth, and IrDA elude conventional sensors but may also carry relevant traffic that must be subjected to ID analysis. In combination with increased reliance on encrypted traffic (such as in the case of IEEE 802.11b or VPN access to protected networks) most network-based IDS are confronted with a partial blinding with regard to relevant traffic. Encrypted VPN and other tunneling mechanisms are also causing topologically oriented firewall systems to lose some of their efficacy since the encapsulated payload only becomes visible at the destination node. A consequence of this observation is

Paper presented at the RTO IST Symposium on "Real Time Intrusion Detection", held in Estoril, Portugal from 27-28 May 2002, and published in RTO-MP-101.

that some successful attack mechanisms (both direct external attacks and attacks induced by Trojan horses) can communicate with external nodes while being invisible to both firewalls and ID sensors since they can tunnel across legitimate traffic or use altogether encrypted traffic that may not get blocked by firewalling mechanisms.

The following sections will discuss mechanisms for addressing these issues, namely the embedding of fine-grained instrumentation particularly into COTS systems and dynamic control mechanisms for the generation of audit data. As the mechanisms discussed here are part of a larger architectural framework, it is also necessary to discuss some underlying assumptions regarding the environment in which the IDS must operate and the threats to be countered.

Architectural Framework

In the framework discussed here, there exist a number of nodes called external reference monitors (ERM) which are repositories for one or more security policies, each presumably derived from a security model. The other component of the framework consists of a number of nodes which are subject to the policies of one or more ERM. The policies obtained from ERM are enforced through externally controlled reference monitors (ECRM) and its enforcement modules (EM). As implied by the term reference monitor, each operation of the controlled nodes is mediated by the ECRM and may only proceed if it is found to be in compliance with all applicable policies.

Applicable policies (and hence the ERMs to be consulted) are determined from the identity of subjects and objects involved which are uniquely identified by the conjunction of a subject identity and a subject type constant. Both constants are embedded in a lattice; an ERM is authorized to issue a policy if and only if it dominates the objects involved in an operation.

Policies are formulated in a formal theory, namely a first order predicate calculus. The subjects and objects are, as noted above, represented by constants which are gathered into sets by predicates. Behavior permitted by a policy is then expressed as additional functions, constants, and predicates. By identifying relations on subjects, objects, and operations and formulating these in the form of predicates one can not only express arbitrary security policies and models within a common, consistent scheme but also use automated deduction mechanisms to derive additional statements and instances. This permits the formulation of a pure security model and the relations and having the deduction mechanism derive all logically valid instances as well as formulating security policies and models in a sufficiently abstract manner such that the complexity of the specifications to be created by human security officers is limited.

It should be noted that the elements of the formal theory do not, besides the use of suggestive names, carry semantic meaning. Rather, for each instance where the mechanism is to be applied (i.e. operating systems in the case of this discussion) there must be an interpretation which supplies the semantics. If one is able to formulate the security models and policies in the form of such abstract statements, an obvious direct consequence of this approach is that the semantics of the security policy enforced are the same regardless of the specific operating system used.

To permit the placement of an upper boundary on the communication complexity imposed by the ERM/ECRM mechanism, each reply to a policy request must contain a lifetime λ specified as an ordered pair of time values. An ECRM or a caching ERM may use a reply without additional policy requests to the issuing ERM for the duration of the lifetime. A special case exists in the case where both elements of λ are equal; this implies that the reply may not be cached at all.

When considering only the operation as a reference monitor, the behavior of the ECRM is reactive. An EM may detect a mediated operation and refer this operation to the ECRM which reformulates the operation in the form of a hypothesis. The ECRM must then consult its locally cached policy information (if any) by attempting to derive the hypothesis from the existing set of policy statements. When successful, the request is granted and may proceed. If no local policy is applicable or sufficient, the ECRM must then pose the hypotheses to all applicable ERM. An ERM may reply with a negative reply tuple (indicating that the hypothesis cannot be deduced from the policy) or a nonempty set of reply tuples which may then in turn be cached by the ECRM for the lifetime of each reply tuple since the replies issued by an ERM may contain the hypothesis posed and be usable in a large number of other mediated requests in addition to the original hypothesis. For a detailed description of the ERM/ECRM mechanism refer to [35].

Enforcement Modules as Instrumentation

As described in the previous section, the ECRM mechanism must mediate all resources controlled by the operating system. In a newly designed system this can be accomplished in a rather straightforward fashion. However, given the need to retain an installed base, particularly complex application programs which may not be available on other platforms, it is of particular interest to provide the benefits of the mechanism outlined above for COTS operating systems and applications.

As described in [33], the enforcement modules can be considered as layered protection mechanisms since some operations occur only as direct consequences at higher abstraction levels which have already been subject to evaluation regarding compliance with applicable policies. The layering mechanism does not need to stop at the edge of the operating system; provided that an application program, middleware component or similar construct concentrates its mediation mechanism and provides the required identification for subjects, objects, and operations any such component can be subsumed under the layered enforcement module framework. In the case of such application layer components the reliability of mediation must, however, be considered since these are unlikely to be part of the trusted computing base.

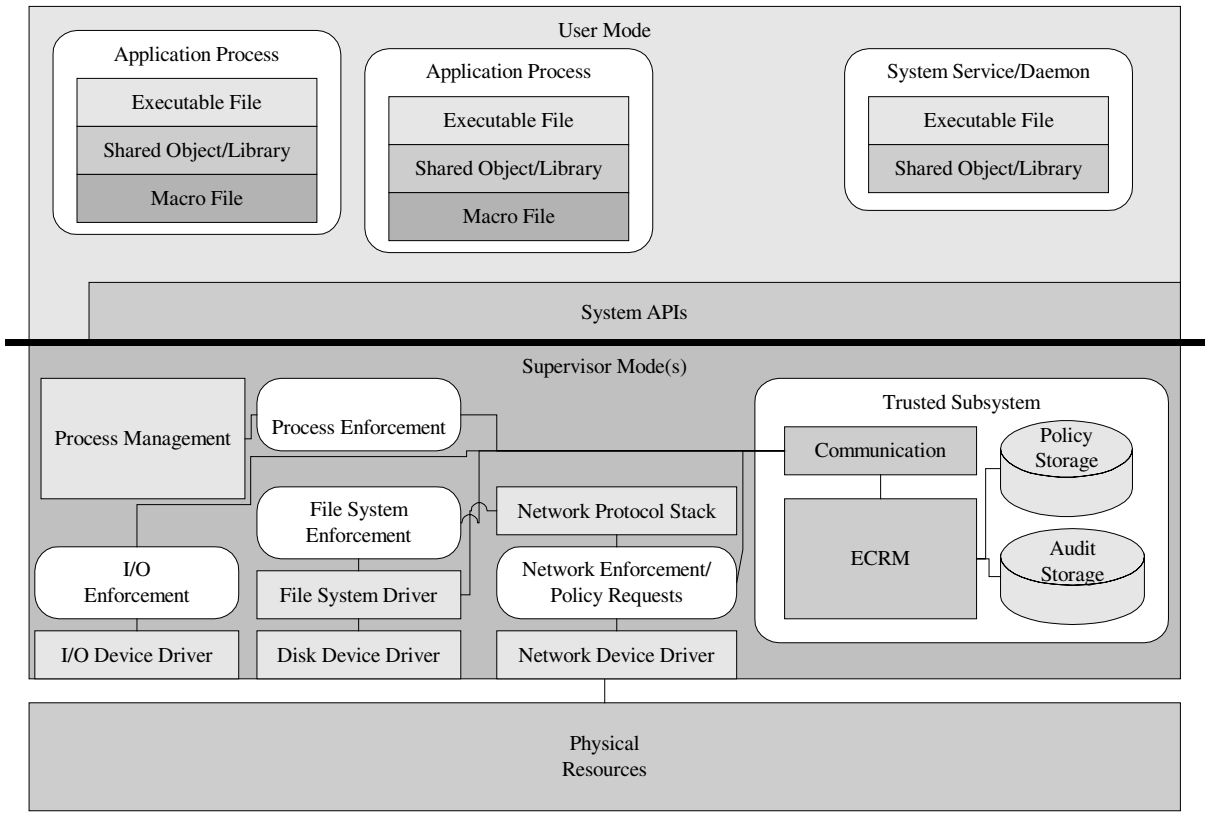


Figure 1: Enforcement Components in an ECRM-Controlled System

The minimum internal enforcement components are shown in figure 1 and will be discussed in the following paragraphs:

I/O Enforcement

The lowest abstraction layer is provided by the physical I/O mechanisms provided by the operating system. These will be used in most cases by higher level subsystems and not accessed directly by application programs. As demonstrated by [22], information on the subsystem accessing a specific I/O mechanism can in and of itself provide valuable information since the observable behavioral pattern for legitimate and normal operations will be of high regularity.

While the mechanisms at this abstraction layer are somewhat limited in their usable scope due to a lack of semantic information, there is other valuable information to be obtained from certain types of I/O mechanisms, for instance in the case of bus-oriented I/O systems the addition or removal of another

entity and, depending on the type of device even a preliminary identification of such entities. This information cannot be obtained as a side effect of reactive behavior in the course of security policy enforcement, rather it must be actively gathered by such modules, the general implementation of which is quite straightforward on both the Microsoft Windows NT and System V Release 4 Unix systems although labeling represents a problem since consistent enumeration of I/O channels even within a single node usually cannot be guaranteed.

In the case of Windows NT, the filter driver mechanism [29,3] can be employed to interposition arbitrary functionality between a device driver and upper layers within the driver stack. Device handling itself is separated into bus drivers handling actual bus operations for devices which must support such topologies and into function drivers dealing with individual devices (or classes of devices) dealing with the device hardware directly or using the primitives supplied by the hardware abstraction layer (HAL).

While Unix derivatives do not provide a mechanism comparable to filter drivers [9,20], a similar mechanism can be provided quite easily given the powerful file abstraction mechanism provided by Unix. Since each device is represented by a (block or character) special file and device drivers are — at least on modern Unix derivatives — present in the form of kernel modules. The entry points provided by these are standardized except for specific I/O control mechanisms.

Regardless of the platform, the issue of memory-mapped I/O must be dealt with either at higher abstraction layers such as the file system or by interacting with the process control mechanism.

File System Enforcement

The file system layer represents a core element of security policy enforcement and ID mechanisms and is closely intertwined with the process (particularly virtual memory and cache management) and network subsystems most modern operating systems which typically also use the file paradigm to model other system elements and hence base their native security models on files.

By interposing a policy-controlled mediation layer enforcing applicable policies a number of important benefits can be realized. In a distributed environment the most important of these is that a policy is enforced equivalently regardless of the configuration of individual nodes or operating systems using the same semantics as defined in the formal theory and using only the subjects, objects, and operations as the basis for policy enforcement. It should be noted that the layer is at a semantically higher layer than the I/O enforcement layer and therefore is capable of both identifying operations and objects at a sufficient level of abstraction and being independent of the implementation of specific file systems. As a result, identification of file objects requires the use of in-band labeling to ensure that the ECRM on all communicating node can properly associate the system-specific file representation with the abstract policy rules. The label itself identifies the object, not its classification, and is used only in policy rule derivation to generate appropriate actions, and never becomes visible to other abstraction layers. The interposition mechanism can also — quite efficiently since the labeling mechanism requires the dynamic adaptation of file sizes and offsets in any case — provide additional transformations on the data being processed such as policy-based mandatory encryption. This becomes particularly relevant since most COTS systems are not adequately protected against accessing storage media using operating systems that do not honor the policies as well as by the emergence of standards such as iSCSI, FCIP, and InfiniBand.

The file system interposition mechanisms second role is in the provision of information to the ECRM for both file-system specific operations as well as for decisions involving other abstraction layers and policy rules which must be satisfied. This is particularly the case if information flow control policies are enforced. The overall control through the ECRM also means that, in addition to being able to audit the individual reactive events described above, audit and ID sensor data can also be obtained when and wherever it is determined to be relevant, possibly even dynamically based on policies and preceding events. One example of such information regarding file objects would be the retrieval of an object label or other information uniquely identifying a file, permitting the incorporation of detailed data when it is deemed to be relevant by the auditing and intrusion detection subsystems.

Implementation of this mechanism under Windows NT is described in [34]; since the file system is built as a set of layers, there exists a mechanism to interpose functionality in the form of file system filter drivers immediately between the I/O manager and the various file systems resulting in attainment of the objectives outlined above. The main drawbacks observed here are that the relevant interfaces are not (if at

all) adequately documented and that for the sake of minor speed benefits dubious multiple code paths exist for the same nominal semantics.

Unix System V Release 4 derivatives, on the contrary do not have a standard mechanism for interposing such functionality; while some extensions such as the SGI IRIX behavior mechanism exist, these are proprietary in nature. However, while not originally intended for this purpose, a mechanism to achieve the desiderata described above exists in the form of the modular virtual file system (VFS) architecture. Designed for adding new file systems without the need for a recompilation of the kernel it introduces a virtualized (albeit undocumented [31,36,20]) interface to file systems which can be used to interpose arbitrary behavior in an enforcement module.

Network Enforcement

The implementation of the Network Protocol Layer is highly performance sensitive and requires a tight integration of all components to reduce critical code path lengths. However, for the purposes of this exposition individual functional components as an in itself layered architecture are described.

For some COTS systems it is necessary to limit the scope of protocol data units (PDUs) they are confronted with since some malformed as well as obscure well-formed PDUs may not be interpreted correctly or cause other problems. An interposition layer therefore must intercept at least inbound PDUs and if necessary modify these before forwarding these to the actual network subsystem. While this role could be assumed by a central firewall mechanism, the computational and memory overhead associated with such operations as well as the need to protect the network stack in end-to-end communication imply that this must take place at each individual node.

The interposed layer must exercise packet and circuit level control regardless of the network protocols used and irrespective of possible encapsulation in end-to-end protocols such as GRE [6] or encryption such as IPSec [14]. This can, in general, also be accomplished only at the end node itself. The enforcement mechanism can grant permission to send and receive PDUs and to establish circuits based on applicable policies. In doing so additional constraints can be levied; one example is the restriction to positive identifed and authenticated peers using confidentiality and integrity protecting communication. The most relevant instance of such a mechanism is in the mandatory use of IPSec to ensure that other nodes in a distributed system also are under the control of ERM nodes although interoperability with non-compliant nodes supporting IPSec can still be maintained as the relevant modifications are only in the requirements and behavior for establishment of key exchange and security associations internal to the node.

As already noted above, the network enforcement layer must interoperate closely with the file system and also with the process enforcement layer to permit the enforcement of some security policies. In some implementations it is also appropriate to insert additional sublayers at higher levels within the network protocol stack for performance reasons; this could e.g. be the case when application-specific layers are incorporated.

An example of an interaction between multiple layers is the problem of tunneling across application layer encryption across network connections; while some of this traffic is legitimate these can also used to encapsulate undesirable traffic and shield this traffic from scrutiny. By permitting only a select number of positively identified application programs (each file of which is identified) and users to use such tools, this problem can at least be partially alleviated.

The collection of audit information can occur at each of the previously identified sublayers and, as above, to an extent that can be modulated dynamically. By interposing dynamic audit and ID data collection at multiple stages within the network data processing the blinding effect due to end-to-end encryption and tunneling can be addressed and additional information on the subjects and objects involved can be used as a basis for ID. Even if a node is communicating with another ECRM node the positive identification of the peer node or nodes in a communication exchange constitutes a potentially valuable data point.

Windows NT implements a number of networking protocols; handling of these protocols is dispersed through several layers which must be instrumented to obtain all necessary information. Most operations, including the insertion of IPSec and firewalling, can occur at the transport driver interface (TDI) layer. This needs to be performed for all network protocols that have to be supported (e.g. IP, DECNet). This has the advantage of capturing all network devices supported by the operating system. For supporting

some features such as the encapsulation of the host stack and embedding a trusted implementation of IPSec, this is insufficient and a filtering mechanism using a wrapping mechanism for the NDIS layer is required, using a manual interception of all communication between NDIS miniport drivers and the surrounding NDIS library. Additionally, not all information necessary for higher-level processing is present at the TDI transport level. For efficiency reasons and to connect various information sources it is necessary to insert another mechanism at the WinSock level, namely a WinSock layered service provider. Details on this mechanism are discussed in [26].

On the contrary, the network subsystem of System V Release 4 Unix derivatives is highly structured; all data traffic is ultimately passed through the asynchronous STREAMS stack even though it may have originated in other API layers such as the traditional BSD socket compatibility subsystem. One can intercept or replace data at multiple sublayers from the link layer upward. Depending on the availability of a native IPSec implementation, either augmentation or wrapping of the existing stack is called for; wrapping is the more general mechanism but implies a considerable performance impact. The STREAMS layer does, however, provide the necessary data structures to easily identify both the process, the executable data performing the operation, and the user, permitting to localize several policy decisions.

Process Enforcement

This abstraction layer can best be viewed as a collection of disparate elements related to the creation and maintenance of processes. This may be somewhat unintuitive depending on the definition of process used; in this context a process is defined as an entity which consists of a collection of resources (e.g. a protected address space, stack frame, register context) and the ability to execute operations. Hence, enforcement involves the use of shared memory segments in addition to process management and must interact closely with other components, particularly the file system and I/O enforcement components which may perform operations relevant particularly to information flow security policies. In terms of auditing and ID, this additionally contributes important information on processes and the identities of other subjects (users, impersonation, application programs) that must be correlated with one another to obtain relevant conclusions. To obtain the necessary semantic information it is necessary to intercept not only the internal object management performed by the operating system but also of parts of the system call interface offered to upper level kernel and unprivileged domain clients.

This necessitates, particularly in the case of Windows NT, a rather unaesthetic implementation in that the relevant interfaces (particularly the Object Manager and the NT Executive Native API) do not anticipate the presence of a filter driver mechanism. One must therefore directly replace the relevant components with a shell providing the same entry points but redirecting the calls on entry and exit to the actual kernel component to perform the necessary operations.

The handling of Unix System V Release 4 derivatives is somewhat similar albeit in a more consistent fashion since central entry point tables exist which can be used to chain in arbitrarily many additional code segments to achieve the purposes described above.

The Role of Intrusion Detectionin a Policy-Controlled Environment

Given the architectural framework outlined previously one must distinguish between several scenarios in which the instrumentation becomes relevant as ID sensors.

For this to occur systematically, one needs to distinguish several scenarios in which different threat models apply; scenarios listed later include the mechanisms of prior scenarios:

Passive Detection

This situation is equivalent to the deployment scenario generally used for IDS, namely intrusion detection decoupled from policy enforcement. The node is fully exposed to all threats from undesirable behavior of insiders as well as to external attacks.

Here the benefits of having extensive instrumentation at the various abstraction layers can be assumed to be roughly equivalent to that provided by [8] and [22] in that they provide a more detailed view of host-based system behavior than what can be expected from native instrumentation typically intended for TCSEC C2 or later equivalents [30,23,13].

For network-based components, the immediate benefit is in the ability to access network PDUs that are subject to end-to-end encapsulation or encryption and in distributing the processing load, addressing bandwidth limitations at central detectors in the process.

In this scenario a policy can be distributed ad hoc and ensuring equivalent behavior across heterogeneous platforms including possible preprocessing before reporting to designated processing and fusion nodes.

Augmented Detection

In this scenario policy enforcement is also decoupled from intrusion detection. Hence, the threat environment is as described in the previous scenario.

However, a dynamic element is introduced at two levels. First, local policy rules can, in response to observations from local sensors, derive additional rules for activation of sensors. This implicitly includes the derivation of intrusion scenarios using the deduction system (which amounts to the use of the deduction engine as a production system) or a secondary anomaly detector capable of generating rules as output. All such rules are implicitly time-bounded using the lifetime mechanism described earlier. The second dynamic element consists of the ECRM nodes sending sensor data or derived hypotheses to ERM nodes which then react to these notifications from one or more nodes. Again, the reaction on the part of the ERM can be induced by the deduction system or an anomaly detector. Policy elements for intrusion detection such as an increase in output volume for certain sensors on specific nodes can thus be obtained and propagated to the ECRM nodes.

It should be noted that the extent to which the sensor activation, output modulation, and subsequent processing by both ERM and ECRM takes place must be weighed carefully since degradation or denial of service can otherwise occur.

Enforcement Augmentation

In this scenario, security policy enforcement is in place and is merely augmented by intrusion detection capabilities; this represents the intended environment for the architecture described in this paper. In the presence of policy enforcement, some premises for the previous scenario are — depending on the policies — no longer applicable.

Policies which enforce that only legitimate operations are performed under a given security model and clearly sets of operations and the circumstances under which these may be performed can a priori eliminate a significant amount of behavior that would otherwise need to be analyzed for signs of intrusion as well as behavior which, while legitimate under the security policy, is abnormal. Some of this ambiguous area for a given set of policies can probably be covered by deduction-based detection; most, however, will need to be analyzed using anomaly detection. Attempts at violating the security policies need to be analyzed in conjunction with legitimate behavior preceding it or concomitant to it; it is mainly in the elimination of some behavior subsequent to initial breaches of policy that would otherwise need to be analyzed forensically.

Attack detection (i.e. operations performed by unauthorized subjects not under the control of policy mechanisms) represent a category which still needs to be addressed. However, given the volume of sensor data and particularly of observations which require further processing, the distributed processing or preprocessing becomes highly valuable even though the amount of sensor data is otherwise reduced.

Fully Reactive

While the drawbacks inherent in this scenario are severe and will presumably preclude it from consideration in most cases, it is included here nonetheless for completeness. This scenario adds definitions to the policy rule sets for rewriting policies to restrict operations normally within the purview of certain subjects or concerning sensitive objects (e.g. limiting access to objects with a given classification on a node for which attacks followed by anomalous behavior of internal subjects has been observed).

Even more so than in the Augmented Detection and Enforcement Augmentation scenarios with reactive sensor and processing behavior this creates severe risks of denial of service for legitimate subjects.

Finally, it is also conceivable that the Fully Reactive scenario is extended through the policy-based activation of active information operations; since this would only be relevant in situations where external attacks are detected this implies that in all likelihood the identity of the adversary cannot be positively verified. This, however, almost ensures that a skilled adversary will be able to use predictable reactive systems against the defender.

THE RÔLE OF SECURE COPROCESSORS

As discussed in [33], the ECRM should be located in a trusted coprocessor subsystem to avoid tampering and to ensure the proper separation from other components particularly if the ECRM is retrofitted onto a COTS system whose assurance may not be adequate in itself given the threat environment. Using a secure coprocessor can ensure that —unless the device itself has been tampered with in which case an attack becomes somewhat obvious — critical audit information is stored in a way so it is inaccessible without considerable effort to an attacker even if he has gained physical access to the system and may delete other relevant data. Another side effect is that the reporting of sensor information will remain accurate even if the kernel of the host operating system itself is compromised. While the sensor data being fed into the coprocessor after such an event are meaningless, the reporting (until it is disrupted) will reliably reflect the state of the system during the attack and thus provide valuable intelligence. Depending on the security policy enforced (e.g. mandatory encryption of all file objects and network transmission using keys not divulged by the secure coprocessors, this also limits the damage that can be inflicted after a system has been compromised to whatever was exposed to nonvolatile storage (i.e. the domain of control of the regular operating system kernel) up to the point where the successful intrusion is discovered and appropriate steps are taken.

Related Work

The subjects discussed in this paper touch on a number of highly active research areas; the following can therefore hardly be anything but an excerpt of some of the more interesting related areas.

The use of production systems (whose behavior loosely resembles what can be obtained through the use of automated deduction techniques) was used in the MIDAS [27] system as well as in the SRI work on intrusion detection beginning with the later stages of IDES [4,18,19].

The use of high resolution host-based sensor information collection has been pursued particularly in the Unix area where system call tracing information is relatively easy and inexpensive to obtain without requiring extensive instrumentation [8,11,32]; this instrumentation method has been used to obtain rules on normal behavior [17] although the use of specification-based constraints imposed prior to monitoring has also been explored [16,28]. The use of intra-kernel call sequences has also been used to realize fine-grained instrumentation, resulting in very high volumes of sensor data [5,22].

The aspect of securing network interfaces by policy-controlled mechanisms inaccessible to other system components is also pursued in [24] although purely embedded solutions also exist [12] while the issue of preventing adversary intelligence gathering through dynamic behavior is discussed in [15].

Conclusions and Future Work

The preceding discussion described several aspects of a security architecture for distributed heterogeneous environments that are relevant to attack and intrusion detection. While the main focus was on the use of instrumentation — particularly through retrofitting of COTS operating systems — at the kernel level critical for the mediation of operations within a system, the remainder of the architecture has considerable side effects that also need exploring. In particular, once a sufficiently stringent set of policies is enforced, at least part of the raison d'être for conventional IDS no longer applies. Instead, one can concentrate on using the sensors for analyzing legitimate but suspect behavior and for attack detection. The overall architecture allows (with the obvious but critical exception of time) the correlation of operations performed by subjects both internal and external to the distributed system through the use of coherent labeling for subjects and objects as well as the description of operations through predicates within the formal theory framework irrespective of the platform an operation is performed on.

Much of the work discussed here is still at a very early stage and at least some of the benefits described are based on extrapolation and conjecture; the immediate task at hand is therefore the validation of the conjectures in a realistic environment. In case of the Enforcement Augmentation approach a significant dependency between the enforcement and intrusion detection policies is expected; quantitative results are therefore necessary to assess the optimum cost-benefit balance between enforcement and (possibly deferred) intrusion detection.

Another aspect that needs to be explored quantitatively is the efficacy of more elaborate anomaly detection algorithms than first moments given the detailed and high volume sensor information available.

Another possible area that can be explored is the integration of the policy-based system with another distributed intrusion detection system such as EMERALD [25]; this would permit an extension of sensor range even if other nodes under surveillance are not equipped with policy-based enforcement mechanisms.

On a larger scale, the instrumentation described in this paper permits the exploration of both deferred and real-time attack and intrusion detection. The latter is particularly significant due to the possible cascading effects of successful intrusions; since similar problems such as multiple-target tracking have confronted the multisensor data fusion community for more than fifty years the application of some techniques developed in this context may yield interesting results.

REFERENCES

1. ANDERSON, J. P. Computer Security Threat Monitoring and Surveillance. Tech. rep., James P Anderson Co., Fort Washington, PA, Apr. 1980.
2. BELLOVIN, S. M. distributed firewalls. *;login: the USENIX Association newsletter 19*, Special Issue on Security (Nov. 1999), 39–47.
3. DEKKER, E. N., AND NEWCOMER, J. M. *Developing Windows NT Device Drivers*. Addison-Wesley, New York, NY, USA, 1999.
4. DENNING, D.E., AND NEUMANN, P.G. Requirements and Model for IDES - A Real-Time Intrusion Detection Expert System. Tech. rep., Computer Science Laboratory, SRI International, Menlo Park, CA, USA, 1985.
5. ELBAUM, S., AND MUNSON, J. C. Intrusion Detection Through Dynamic Software Measurement. In *Proceedings of the Workshop on Intrusion Detection and Network Monitoring* (Santa Clara, CA, USA, Apr. 1999), USENIX Association, pp. 41–50.
6. FARINACCI, D., LI, T., HANKS, S., MAYER, D., AND TRAINA, P. RFC 2784: Generic Routing Encapsulation, Mar. 2000. Status: PROPOSED STANDARD.
7. FORREST, S., HOFMEYR, S. A., AND SOMAYAJI, A. Computer Immunology. *Communications of the ACM 40*, 10 (Oct. 1997), 88–96.
8. FORREST, S., HOFMEYR, S. A., SOMAYAJI, A., AND LONGSTAFF, T. A. A Sense of Self for Unix Processes. In *Proceedings of the 1996 IEEE Symposium on Security and Privacy (SOSP '96)* (Oakland, CA, USA, May 1996), IEEE Computer Society Press, pp. 120–128.
9. GOODHEART, B., AND COX, J. *The Magic Garden Explained: The Internals of Unix System V Release 4*. Prentice Hall, Englewood Cliffs, NJ, USA, 1994.
10. HALME, L. R., AND BAUER, R. K. AINT misbehaving – A taxonomy of anti-intrusion techniques. In *Proceedings of the 18th NIST- NCSC National Information Systems Security Conference* (1995), pp. 163–172.
11. HELMER, G. G., WONG, J. S. K., HONAVAR, V., AND MILLER, L. Intelligent Agents for Intrusion Detection. In *Proceedings of the 1998 IEEE Information Technology Conference* (Syracuse, NY, USA, Sept. 1998), IEEE Computer Society Press, pp. 121–124.
12. IOANNIDIS, S., KEROMYTIS, A. D., BELLOVIN, S. M., AND SMITH, J. M. Implementing a Distributed Firewall. In *Proceedings of the 7th ACM Conference on Computer and Communications Security*

(CCS-2000) (Athens, Greece, Nov. 2000), S. Jajodia and P. Samarati, Eds., Association for Computing Machinery, pp. 190–199.

13. ISO/IEC STANDARD 15408. Common Criteria for Information Technology Security Evaluation, version 2.1, Dec. 1999.

14. KENT, S., AND ATKINSON, R. RFC 2401: Security Architecture for the Internet Protocol, Nov. 1998. Status: PROPOSED STANDARD.

15. KEWLEY, D., FINK, R., LOWRY, J., AND DEAN, M. Dynamic Approaches to Thwart Adversary Intelligence Gathering. In *Proceedings DARPA Information Survivability Conference & Exposition II (DISCEX '01)* (Anaheim, CA, USA, June 2001), IEEE Computer Society Press, pp. 329–336.

16. KO, C., FINK, G., AND LEVITT, K. Automated Detection of Vulnerabilities in Priviledged Programs by Execution Monitoring. In *Proceedings 10th Annual Computer Security Applications Conference (ACSAC'94)* (Orlando, FL, USA, Dec. 1994), IEEE Computer Society Press, pp. 134–144.

17. LEE, W., STOLFO, S. J., AND CHAN, P. K. Learning Patterns from UNIX Process Execution Traces for Intrusion Detection. In *Proceedings of the AAAI Workshop on AI Approaches to Fraud Detection and Risk Management* (Providence, RI, USA, July 1997), AAAI Press, pp. 50–56.

18. LUNT, T. F., JAGANNATHAN, R., LEE, R., LISTGARTEN, S., EDWARDS, D. L., NEUMANN, P. G., JAVITZ, H. S., AND VALDES, A. Development and Application of IDES: A Real-Time Intrusion-Detection Expert System. Tech. rep., Computer Science Laboratory, SRI International, Menlo Park, CA, USA, 1988.

19. LUNT, T. F., TAMARU, A., GILHAM, F., JAGANNATHAN, R., NEUMANN, P. G., JAVITZ, H. S., VALDES, A., AND GARVEY, T. D. A Real-Time Intrusion Detection Expert System (IDES) – Final Technical Report. Tech. rep., Computer Science Laboratory, SRI International, Menlo Park, CA, USA, 1992.

20. MAURO, J., AND MCDOUGALL, R. *Solaris Internals: Core Kernel Architecture, Vol. 1.* Prentice Hall, Englewood Cliffs, NJ, USA, 2000.

21. MÉ, L. Security Audit Trail Analysis Using Genetic Algorithms. In *Proceedings of the 12th International Conference on Computer Safety, Reliability and Security (SAFECOMP 93)* (Poznan-Kiekrz, Poland, Oct. 1993), pp. 329–340.

22. MUNSON, J. C., AND WIMER, S. Watcher: The Missing Piece of the Security Puzzle. In *Proceedings 17th Annual Computer Security Applications Conference (ACSAC'01)* (New Orleans, LA, USA, Dec. 2001), IEEE Computer Society Press, pp. 230–239.

23. NATIONAL SECURITY AGENCY INFORMATION SYSTEMS SECURITY ORGANIZATION. *Controlled Access Protection Profile*. Fort George G. Meade, MD, USA, Oct. 1999. Version 1.D.

24. PAYNE, C., AND MARKHAM, T. Architecture and Applications for a Distributed Embedded Firewall. In *Proceedings 17th Annual Computer Security Applications Conference (ACSAC'01)* (New Orleans, LA, USA, Dec. 2001), IEEE Computer Society Press, pp. 329–336.

25. PORRAS, P. A., AND NEUMANN, P. G. EMERALD: Event Monitoring Enabling Responses to Anomalous Live Disturbances. In *Proceedings of the 20th NIST-NCSC National Information Systems Security Conference* (1997), pp. 353–365.

26. RADEMER, E., AND WOLTHUSEN, S. Transparent Access To Encrypted Data Using Operating System Network Stack Extensions. In *Communications and Multimedia Security Issues of the New Century: Proceedings of the IFIP TC6/TC11 Fifth Joint Working Conference on Communications and Multimedia Security (CMS'01)* (Darmstadt, Germany, May 2001), R. Steinmetz, J. Dittman, and M. Steinebach, Eds., IFIP, Kluwer Academic Publishers, pp. 213–226.

27. SEBRING, M. M., SHELLHOUSE, E., HANNA, M. E., AND WHITEHURST, R. A. Expert System in Intrusion Detection: A Case Study. In *Proceedings of the 11th National Computer Security Conference* (1988), pp. 74–81.

28. SEKAR, R., BOWEN, T., AND SEGAL, M. On Preventing Intrusions by Process Behavior Monitoring. In *Proceedings of the Workshop on Intrusion Detection and Network Monitoring* (Santa Clara, CA, USA, Apr. 1999), USENIX Association, pp. 29–40.

29. SOLOMON, D., AND RUSSINOVICH, M. *Inside Windows 2000*, 3rd ed. Microsoft Press, Bellevue, WA, USA, 2000.

30. UNITED STATES DEPARTMENT OF DEFENSE. *DoD 5200.28-STD: Department of Defense (DoD) Trusted Computer System Evaluation Criteria (TCSEC)*, 1985.

31. VAHALIA, U. *UNIX Internals: The New Frontiers*. Prentice Hall, Englewood Cliffs, NJ, USA, 1995.

32. WARRENDER, C., FORREST, S., AND PEARLMUTTER, B. Detecting Intrusions Using System Calls: Alternative Data Models. In *Proceedings of the 1996 IEEE Symposium on Security and Privacy (SOSP '96)* (Oakland, CA, USA, May 1996), IEEE Computer Society Press, pp. 133–145.

33. WOLTHUSEN, S. Layered multipoint network defense and security policy enforcement. In *Proceedings from the Second Annual IEEE SMC Information Assurance Workshop, United States Military Academy* (West Point, NY, USA, June 2001), IEEE Press, pp. 100–108.

34. WOLTHUSEN, S. Security Policy Enforcement at the File System Level in the Windows NT Operating System Family. In *Proceedings 17th Annual Computer Security Applications Conference (ACSAC'01)* (New Orleans, LA, USA, Dec. 2001), IEEE Computer Society Press, pp. 55–63.

35. WOLTHUSEN, S. Access and Use Control using Externally Controlled Reference Monitors. *ACM Operating Systems Review* 36, 1 (2002), 58–69.

36. ZADOK, E., BADULESCU, I., AND SHENDER, A. Extending File Systems Using Stackable Templates. In *Proceedings of the USENIX 1999 Annual Technical Conference* (Berkeley, CA, USA, June 1999), USENIX Association, pp. 57–70.

This page has been deliberately left blank

Page intentionnellement blanche

Click here to view PowerPoint presentation; Press Esc to exit

CRIM: An Approach to Correlate Alerts and Recognize Malicious Intentions

Frédéric Cuppens, Alexandre Miège
ONERA Centre de Toulouse
2, av. Edouard Belin
31055, Toulouse CEDEX, France

Abstract

This paper presents the work we have done within the MIRADOR project to design CRIM, a cooperative module for intrusion detection systems (IDS). This module implements functions to manage, cluster, merge and correlate alerts. The clustering and merging functions recognize alerts that correspond to the same occurrence of an attack and create a new alert that merge data contained in these various alerts. Experiments show that these functions significantly reduce the number of alerts. However, we also observe that alerts we obtain are still too elementary to be managed by a security administrator. The purpose of the correlation function is thus to generate global and synthetic alerts. This paper focuses on the approach we suggest to design this function.

1. Introduction

There are actually two main intrusion detection approaches: the behavioral approach (also called *anomaly detection*) and the signature analysis (also called *misuse detection*). Anomaly detection is based on statistical description of the normal behavior of users or applications. The objective is then to detect any abnormal action performed by these users or applications. The second approach, called misuse detection, is based on collecting attack signatures in order to store them in an attack base. The IDS then parses audit files to find patterns that match the description of an attack stored in the attack base.

None of these approaches is fully satisfactory. They generally generate many false positives (corresponding to a false alert), false negatives (corresponding to a non-detected attack) and the alerts are too elementary and not enough accurate to be directly managed by a security administrator.

For instance, anomaly detection can generate many false positives. This is because deviation from normal behavior does not always correspond to the occurrence of an attack. Moreover, a malicious internal user can *slowly* modify his behavior so that the final behavior includes an attack. The IDS will learn this new behavior and will associate it with a normal behavior. Therefore, the attack will not be detected. This corresponds to the occurrence of a false negative.

The problem of exhaustively defining the attack base is a major difficulty of misuse detection. Therefore, misuse detection can also generate many false negatives especially when a given attack has many close but different implementations. Moreover, in current products, the quality of signatures expressed in the attack base is generally not sufficient to avoid false positives.

In this context, a promising approach is to develop a cooperation module between several IDS to analyze alerts and generate more global and synthetic alerts. The objective is to reduce the number of generated alerts and increase the detection rate of attacks. We also want to provide the security administrator with alerts that can be used to take the right decision.

CRIM is such a cooperative module we have developed within MIRADOR. MIRADOR is a project initiated by the French Defense Agency (DGA) and is led by Alcatel in collaboration with 3 research laboratories: ONERA, ENST-Bretagne and Supelec. MIRADOR aims to build a cooperative and adaptive IDS platform. CRIM is part of this project and implements the following functions: alert clustering, alert merging and alert correlation. A cluster of alerts is a set of alerts that correspond to the same occurrence of an attack. The purpose of the merging function is then to create a new alert that is representative of the information contained in the various alerts belonging to this cluster.

Paper presented at the RTO IST Symposium on "Real Time Intrusion Detection", held in Estoril, Portugal from 27-28 May 2002, and published in RTO-MP-101.

This approach enables us to reduce the number of alerts transmitted to the security administrator. We check our merging function over an attack base of 87 “elementary” attacks. An elementary attack corresponds to a non-decomposable step of a given scenario. For this experiment, we used two different network-based IDS: Snort [Roe99] and e-Trust [CA00]. The results we obtained were as follows. The 87 attacks generated 325 alerts: 264 for Snort and 61 for e-Trust. Only 69 attacks were detected: 41 by both Snort and e-Trust, 27 by Snort but not by e-Trust, 1 by e-Trust but not by Snort and 18 attacks were not detected. When checking our clustering function on the above attack base, we actually obtained 101 clusters.

But, the alerts we obtained still correspond to too elementary alerts. The consequence will be that the security administrator will have difficulty to take the correct decision when receiving these alerts.

Therefore, a complementary analysis must be performed. This is the purpose of the correlation function. The principle of the correlation function is to consider that the intruder wants to achieve a malicious objective but he cannot generally get his way by only performing a single attack. Instead, he usually performs several attacks that correspond to steps of a more global intrusion plan that enables him to achieve his malicious objective. Notice that we include, in the intrusion plan, preliminary steps the intruder generally performs to collect various information on configuration of the system to be attacked.

Classical IDS only detect elementary attacks that correspond to the steps of this intrusion plan. The objective of the correlation function is thus to correlate alerts in order to recognize the intrusion plan that is currently executed by the intruder.

In this paper, we present the approach we suggest implementing the correlation function. The remainder of this paper is organized as follows. Section 1 summarizes the main principles of our approach. We first introduce the architecture of CRIM, the cooperative module we developed for intrusion detection. We shortly presents the objectives of the clustering, merging and correlation functions. We then suggest our approach to modeling alerts and attacks. Both models are based on first order logic and are used in our correlation approach. Actually, our representation of attacks is based on the LAMBDA language [CO00]. Section 3 sketches our correlation approach, comparing it with other approaches suggested in the literature. Section 4 formalizes this approach and section 5 further refines it by introducing the concept of abductive correlation. Finally, section 6 concludes this paper.

2. General principles

2.1. CRIM Architecture

Figure 1 presents the main principles we suggest developing a cooperation module for intrusion detection. There are six main functions in this module.

The alert base management function receives the alerts generated by different IDS and stores them for further analysis by the cooperation module. We shall assume that all these alerts are compliant with the Intrusion Detection Message Exchange Format (IDMEF) [CD01]. The purpose of the IDMEF is to define common data formats and exchange procedures for sharing information of interest to intrusion detection and response systems and those that may need to interact with them.

The approach we suggest to implement the alert base management function is to convert the IDMEF messages into a set of tuples and to store them into a relational database (see section 2.2 below).

The clustering function can then have an access to this database and generates clusters of alerts. When an attack occurs, the IDS connected to CRIM may generate several alerts for this attack. The clustering function attempts to recognize the alerts that actually correspond to the same occurrence of an attack. These alerts are brought into a cluster. As presented in [Cup01], a relation of similarity connects alerts belonging to the same cluster. In our implementation, we consider that two alerts are similar if their classification, detection time, source and target are similar. We shortly sketch our approach to define similarity relations between these four attributes.

The classification name is used by each IDS to identify an attack. Therefore, we can consider that two alerts might correspond to the same attack if their classification are identical. However, classification names are often vendor-specific and may differ from one IDS to another. This might complicate the clustering task. To solve this problem, the similarity function can access a table managed by CRIM that provides correspondence between classification names generated by each IDS.

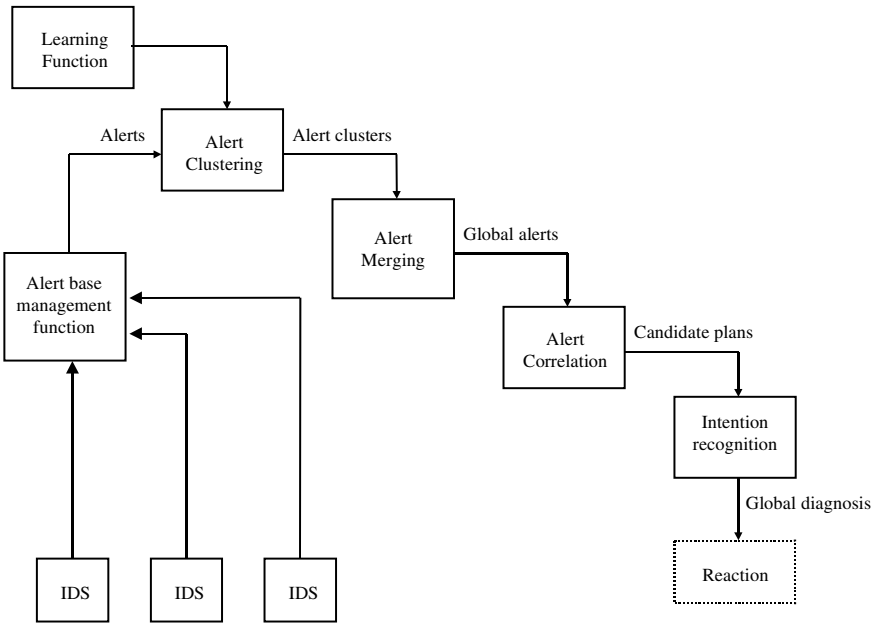


Figure 1: CRIM architecture

When alert classifications are similar, next step is then to compare their detection time. The approach to cluster detection time is to define a delay: two alerts will not be considered similar if the temporal difference between their detection time exceeds this delay. We notice that properly calibrating this delay is crucial to obtain good clustering results (see [Cup01] for further details).

Finally, last criteria to cluster alerts is to compare their source and target. The approach to define similarity between these attributes is to specify, for each attack classification, the expected similarity requirement for the source and target. Sources and targets are both described by a node, a user, a process and a service. This is why a similarity requirement is a list of attributes included in [node, service, user, process]. For most attacks, we decided to consider that two sources (resp. two targets) are similar if both the nodes and services of the sources (resp. of the targets) are similar. In this case, the similarity requirement is expressed by [node, service]. However, for some attacks (for instance a tcp-scan) we specified that it was sufficient to have similar node addresses to have similar targets. Hence, for these attacks, the similarity requirement for the target is expressed by [node]. And, if the source of the attack may be spoofed, two alerts are considered similar even though the sources are completely different. In this case, the similarity requirement for the source is [ ].

Experimentation showed that tuning the clustering function in order to have adequate definitions of similarity relations is quite tedious to perform manually. This is the reason why we have developed a learning function that automatically parameterized various similarity relations. The approach of this function consists in launching the set of attacks to be learned and collecting for each attack the set of alerts generated by all the IDS connected to the platform. The learning function then analyses each set of alerts and automatically defines similarity relations for classification (by creating a table of correspondence between classification names), for detection time (by deriving a delay) and for source and target (by creating the similarity requirement).

Each cluster is then sent to the alert merging function. This function was also presented in [Cup01]. For each cluster, this function creates a new global alert that is representative of the information contained in the various alerts belonging to this cluster. The general process to derive a global alert for each cluster is the following. Let us assume that a new alert $alert_i$ is inserted. There are two possibilities. (1) If there is no alert similar to $alert_i$, then a new cluster is created and a global alert is derived for this cluster. Of course, this global alert will be quite similar to $alert_i$. (2) If $alert_i$ might be inserted in an already existing cluster, then the global alert associated with this cluster is updated by merging $alert_i$ with the global alert of this cluster. Generating the global alert consists in "gathering" most of the information specified in the alerts. This is represented by rules specifying how the global alert is derived from the description of alerts belonging to the cluster (see [Cup01] for further details).

The purpose of this paper is to present next step in our cooperation module, that is the correlation function. This function further analyzes the cluster alerts provided as outputs by the merging function. As mentioned in the introduction, we observe that the merging function generally provides too elementary alerts. The objective

of the correlation function is thus to correlate alerts in order to provide the security administrator with more synthetic information.

The result of the correlation function is a set of candidate plans that correspond to the intrusion under execution by the intruder. However, the final objective of the intruder is perhaps not achieved yet. Next step in our cooperation module is thus to develop the intention recognition function. The purpose of this function is to extrapolate these candidate plans in order to anticipate the intruder actions. This function should provide a global diagnosis of the past (what the intruder has performed up to now), the present (what the intruder has obtained and what is the current security state of the system targeted by the intruder) and the future (how the intruder will go on). The result of this function is to be used by the reaction function to help the system administrator to choose the best counter measure to be launched to prevent the malicious actions performed by the intruder.

As mentioned in the introduction, we shall only present in this paper our approach to correlating alerts. The intention recognition function is not presented. It is under development and the approach we suggest for this function is briefly sketched in the conclusion of this paper.

2.2. Alert modeling

In our approach, every alert is modeled using the IDMEF format. A Document Type Definition (DTD) has been proposed to describe IDMEF data format through XML documents. This is the representation we shall consider in the remainder of this paper.

However, the correlation function does not directly deal with this XML representation of alerts. It is actually automatically converted into a set of logical facts. This set of facts is then stored in a database.

For instance, let us consider the following portion of an alert in the IDMEF format:

```
<?xml version="1.0"?>
<!DOCTYPE IDMEF-Message PUBLIC "-//IETF//DTD RFCxxxx IDMEF v0.3//EN" "/usr/sbin/idmef-message.dtd">
<IDMEF-Message version="0.3">
<Alert ident="249">
        <Analyzer analyzerid="snort-0000-zp109">
                <Node category="dns">
                        <location>unknown</location>
                        <name>zp109</name>
                </Node>
                <Process>
                        <name>snort</name>
                </Process>
        </Analyzer>
….
</Alert>
</IDMEF-Message>
```

It is translated into the following logical representation (where “,” represents conjunction):

```
alert(1),
        ident(1,”249”),
        analyzer(1,2),
                analyzerid(2, “snort-0000-zp109"),
                analyzer_node(2,3),
                        node_category(3,”dns”),
                        node_location(3,”unknown”),
                        node_name(3,”zp109”),
                analyzer_process(2,4),
                        process_name(4,”snort”),
        ….
```

In this representation, ident, analyzer, analyzerid, etc. are binary predicates we use to represent the alert. Predicates are also introduced to describe other portions of an alert description such as detect time, create time, source, target, classification, etc. Actually, we have defined 34 predicates to completely represent all the possible fields of an alert as suggested in the IDMEF format. Numbers 1, 2, 3, 4, … that appears in the above description correspond to object identifiers that are internally created to represent all the sub-parts of an alert description.

2.3. Attack specification in LAMBDA

Since our approach to alert correlation is based on attacks specified in the LAMBDA language, we shortly recall the main principles of this language (see also [CO00] for a more detailed presentation).

In this language, an attack is specified using five fields:

- Attack Pre-condition: A logical condition that specifies the conditions to be satisfied for the attack to succeed.
- Attack Post-condition: A logical condition that specifies the effect of the attack when this attack succeeds.
- Attack scenario: The combination of events the intruder performs when executing the attack.
- Detection scenario: The combination of events that are necessary to detect an occurrence of the attack.
- Verification scenario: A combination of events to be launched to check if the attack succeeds.

Notice that other fields might be included in the attack description. For instance, the ADELE language [MM01] suggests introducing a "reaction" field to specify the actions to be launched when the attack is detected. Actually, in the remainder of this paper, we shall only consider the "Pre-condition", "Post-condition" and "Detection scenario" fields.

The pre-condition and post-condition of an attack correspond to description of conditions over the *system's state*. For this purpose, we use a language, denoted L_1, which is based on the logic of predicates. Predicates are used to describe properties of the state relevant to the description of an attack. The set of predicates used to represent these state conditions is partly inspired from the taxonomy suggested by the Darpa to classify attacks (see [Ken99] for a complete presentation of this taxonomy). More specifically, we shall use:

- A predicate to specify the access level of the intruder over the target system: access_level. For example, the fact access_level(bad_guy,192.168.12.3,local) specifies that the user whose name is "bad_guy" has a local access to host 192.168.12.3. Possible values of the access level are remote, local, user, root and physical.
- A set of predicates to specify the effects of attacks on the target system. This set includes predicates deny_of_service, alter and (illegal) use. For instance, the fact deny_of_service(192.168.12.3) specifies that the attack causes a deny of service on host 192.168.12.3.
- Predicates to specify conditions on the state of the source or target systems. For instance use_service(192.168.12.3,showmount) specifies that service showmount is active on host 192.168.12.3.

These predicates are combined using the logical connectives "," (conjunction denoted by a comma) and "not" (negation). Currently, we do not allow using disjunction in the logical description of an attack. Another restriction is that negation only applies to predicates, not to conjunctive expressions.

Figure 2 provides 4 examples of attacks specified in LAMBDA: NFS mount, Modification of .rhost file, TCPScan and Winnuke. In this description, terms starting with an upper case letter correspond to variables and other terms correspond to constants. For instance, pre-condition of NFS mount attack says:

- access_level(Source_user,Target_address,remote),mounted_partition(Target_address,Partition)

  that is, to perform NFS mount attack, the intruder Source_user must have a remote access on the target whose IP address is Target_address and Partition must be a mounted partition.

The post condition of this attack says:

- can_access(Source_user,Partition)

  that is the intruder Source_user gets an access on the mounted partition Partition.

Notice that sometimes the effect of an attack is simply a knowledge gain for the attacker about the target system. This is for instance the case of attack TCPScan in figure 2. Describing this kind of attacks is very important since their occurrence often corresponds to preliminary steps of a more global attack scenario. In order to represent a knowledge gain, we extend language L_1 so that it also includes a meta-predicate (actually a logical modality) *knows*. For instance, if bad_guy is the attacker, then knows(bad_guy, use_service(192.168.12.3,'NetBios')) means that bad_guy knows that *NetBios* is an active service of system whose IP address is 192.168.12.3.

The other fields of an attack description in LAMBDA correspond to attack scenario, detection scenario and verification scenario[1]. These scenarios are specified using event calculus algebra. This algebra enables us to combine several events using operators such as: ; (sequential composition), | (parallel unconstrained execution), ? (non deterministic choice), & (synchronized execution) and if_not (exclusion of an event when another event occurs). However, all the examples of attacks we shall use in this paper (including examples presented in figure 2) actually correspond to elementary scenarios based on a single event. This is represented by:

- <scenario>Action</scenario>

  to specify that Action is the single event corresponding to the attack scenario.

- <detection>Alert</detection>

  to specify that Alert is the single event corresponding to the detection of the attack.

Finally, conditions appearing in fields cond_scenario and cond_detection are used to formulate description of the event specified in the scenario and detection fields. The cond_scenario field is generally specified using the *script* predicate to represent the command the intruder runs to perform the attack. The cond_detection field is used to describe the main attributes of the alert we expect when the attack occurs. This corresponds to a logical expression without restriction (that is, it can include conjunction, disjunction or negation). It is built using the predicates we introduced in the previous section to logically model an alert. For instance, expression:

- alert(Alert), classification(Alert,"MIR-0163"), source(Alert,Source), source_user(Source,Source_user)

specifies that Alert is an alert whose classification is "MIR-0163" and source must match a given variable Source. The user associated with this source is another variable Source_user. This description enables us to formulate constraints between the various fields of an alert and the variables used in the pre_condition and post_condition description of an attack.

Notice that, in the following, the alert classification will have always the form "MIR-xxxx". This corresponds to an internal classification of attacks used within the MIRADOR project. This classification is used by the merging function to translate "vendor specific" classification into a common classification so that it is thus possible to make correspondence between alert classifications generated by two different IDS (see [Cup01] for more details).

[1] Actually, description of the verification_scenario field is not provided in the examples of figure 2.

```
<?xml version="1.0" encoding="UTF-8"?>
<attack attackid="MIR-0163">

 <name>mount partition</name>

 <pre>access_level(Source_user,Target_address,remote),
       mounted_partition(Target_address,Partition)
 </pre>

 <post>can_access(Source_user,Partition)
 </post>

 <scenario>Action</scenario>
 <cond_scenario>
  script(Action,'mount -t nfs $Partition:$Target_address $Partition')
 </cond_scenario>

 <detection>Alert</detection>
 <cond_detection>alert(Alert),
                      source(Alert,Source),
                      source_user(Source,Source_user),
                       target(Alert,Target),
                       target_node(Target,Target_node),
                      address(Target_node,Target_address),
                      classification(Alert,"MIR-0163")
 </cond_detection>

</attack>
```

Lambda attack MIR-0163 – NFS Mount

```
<?xml version="1.0" encoding="UTF-8"?>
<attack attackid="MIR-0164">

 <name>modification du .rhost</name>

 <pre> access_level(Source_user,Target_address,remote),
         can_access(Source_user,Partition),
         owner(Partition, Target_user),
         userid(Target_user,Target_address,Userid)
 </pre>

 <post> access_level(Source_user,Target_address,user)
 </post>

 <scenario>Action</scenario>
 <cond_scenario>script(Action,'cat "++" > .rhost')</cond_scenario>

 <detection>Alert</detection>
 <cond_detection>alert(Alert),
                      source(Alert,Source),
                      source_user(Source,Source_user),
                       target(Alert,Target),
                       target_node(Target,Target_node),
                       address(Target_node,Target_address),
                       target_user(Target,Target_user),
                       classification(Alert,"MIR-0164")
 </cond_detection>

</attack>
```

Lambda attack MIR-0164 – Modification of .rhost

```
<?xml version="1.0" encoding="UTF-8"?>
<attack attackid="MIR-0073">

<name>tcpscan sur la cible</name>

<pre>use_soft(Source_address,tcpscan),
    use_service(Target_address,Target_service),
    service_type(Target_service,tcp)
</pre>
<post>
 knows(Source_user,use_service(Target_address,Target_service))
</post>

<scenario>Action</scenario>
<cond_scenario>
 script(Action,'tcpscan $Target_address')
</cond_scenario>

<detection>Alert</detection>
<cond_detection>alert(Alert),
                     source(Alert,Source),
                     source_node(Source,Source_node),
                     address(Source_node,Source_address),
                     source_user(Source,Source_user),
                     target(Alert,Target),
                     target_node(Target,Target_node),
                     address(Target_node,Target_address),
                     target_service(Target,Target_service),
                     classification(Alert,"MIR-0073")
</cond_detection>

</attack>
```

Lambda attack MIR-0073 – TCPScan

```
<?xml version="1.0" encoding="UTF-8"?>
<attack attackid="MIR-0036">

<name>winnuke sur la cible</name>

<pre>use_os(Target_address,windows),
      state(Target_address,available)
</pre>

<post>
 deny_of_service(Target_address)
</post>

<scenario>Action</scenario>
<cond_scenario>script(Action,'winnuke $Target_address')
</cond_scenario>

<detection>Alert</detection>
<cond_detection>alert(Alert),
                     source(Alert,Source),
                     source_node(Source,Source_node),
                     address(Source_node,Source_address),
                     target(Alert,Target),target_node(Target,Target_node),
                     address(Target_node,Target_address),
                     classification(Alert,"MIR-0036")
</cond_detection>

</attack>
```

Lambda attaque MIR-0036 – Winnuke

Figure 2: Attack specification in Lambda

3. Explicit correlation

We identified two main approaches to achieve correlation[2]:

- Explicit correlation of events is used when the security administrator is able to express some connection between events that he knows. This connection may be a logical link based on knowledge of relations between alerts. It may be also a link depending on the topology of information system's components [Hua98,DW01].
- Implicit correlation of events is used when data analysis brings out some mappings (may be statistical ones) and relations between events. This approach is mainly based on observing groups of alerts and extracting implicit relations between them. Many works show that intrusion detection probes produce groups of alerts according to the configuration data, the traffic and the topology of information system under surveillance. Such approaches are based on learning techniques (classification [Lee99], data mining [Zer99], neural network [Lip99], …) and should significantly reduce the amount of alerts we have to deal with.

We opt for the explicit approach to carry out the correlation function. Thus, it must be possible to express explicitly known logical links between attacks. This is achieved by using the following predicate:

- attack_correlation(Attack1,Attack2): this predicate says that Attack1 may be correlated with Attack2, that is Attack1 enables the intruder to then perform Attack2.

For instance, the fact attack_correlation("MIR-0066","MIR-0162") specifies that it is possible to correlate attack "MIR-0066" (which corresponds to "rpcinfo") with attack "MIR-0162" (which corresponds to "showmount"). This is because the attack "rpcinfo" enables the intruder to learn if rpc service "showmount" is active.

However, our objective is not to correlate attacks but to correlate alerts. Predicate "attack_correlation" is too coarse for this purpose because it does not provide conditions the alerts must satisfy to correlate them. For instance, we can only consider that alerts corresponding to attacks "MIR-0066" and "MIR-0162" are steps of the same scenario if the target systems appearing in these alerts are equal. If this condition is not satisfied, there is no reason to correlate these alerts.

Therefore, we also introduce the following predicate:

- alert_correlation(Alert1,Alert2): this predicate says that Alert1 may be correlated with Alert2.

This predicate is used to specify *correlation rules*. The conclusion of a correlation rule has always the form alert_correlation(Alert1,Alert2). Its premise specifies the conditions Alert1 and Alert2 must satisfy to conclude that Alert1 and Alert2 can be correlated as part of a given attack scenario.

For instance, figure 3 presents the rule to correlate two alerts whose classifications are respectively "MIR-0066" and "MIR-0162". The premise of a correlation rule has always three parts. Part 1 and 2 respectively provide a description of the two alerts to be correlated. These descriptions correspond to logical conditions expressed in the language presented in section 2.2 to model alerts. Part 3 of the premise expresses the conditions to be satisfied to correlate the two alerts. In the above example, there are two such conditions:

- Condition 1 says that the target addresses appearing in the alerts must be equal (meaning that attacks rpcinfo and showmount applies to the same target)
- Condition 2 says that the service name appearing in Alert1 must be equal to service "mountd" (meaning that one of the services provided by rpcinfo is equal to "mountd")

Notice that there is always an implicit condition to correlate two alerts Alert1 and Alert2. This condition says that Alert1 must occur before Alert2. It is checked by comparing the detect time of Alert1 with the detect time of Alert2. However, this condition is not directly expressed in the correlation rules but systematically checked when the correlation rules are evaluated (see section 4.5 for more details on application of correlation rules).

As the reader may notice, specifying correlation rules would be a quite complex task to perform manually:

- It would be tedious for the administrator at least from a syntactical point of view.

---

[2] Notice that several authors use the terms "alert correlation" for functions that actually correspond to "alert merging" in our terminology (see [VS01] for instance). We do not consider such approaches in the remainder of this paper.

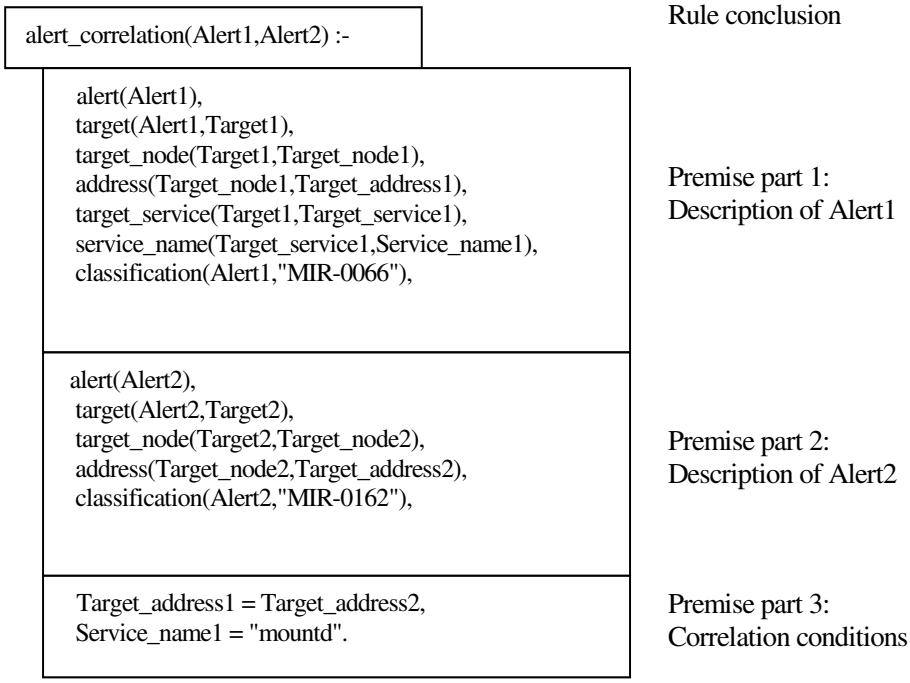


Figure 3: Example of correlation rule
between alerts corresponding to attacks “MIR-0066” (rpcinfo) and “MIR-162” (showmount)

- It is not obvious to be exhaustive, that is not to forget correlation rules specifying pairs of alerts that may be correlated.
- It is also not always obvious to specify the right correlation conditions.

This is the reason why it would be very interesting to have a method to automatically generate correlation rules. We have developed such a method, called semi-explicit correlation. It is presented in the following section.

4. Semi-explicit correlation in LAMBDA

4.1. Background of the approach

This section presents the approach we suggest to performing alert correlation. This approach is based on the analysis of attack description specified in LAMBDA. The central idea of the approach is to recognize whether executing a given attack can *contribute* to execute another attack.

This idea is modeled by specifying possible logical links between the post-condition of an attack A and the pre-condition of an attack B. If such a link exists, then it is possible to correlate an occurrence of attack A with an occurrence of attack B because we can assume that the intruder has performed A as a step that enables him to perform B.

To formally define this kind of correlation between the post-condition of an attack and the pre-condition of another attack, let post(A) be the logical formula representing the post-condition of attack A and pre(B) be the logical formula representing the pre-condition of attack B. Of course, we can correlate attack A and attack B if post(A) logically implies pre(B), that is:

$$post(A) \rightarrow pre(B)$$

However, this definition is generally too strong. This is because it is sufficient to correlate attack A with attack B that attack A “contributes” to the realization of attack B. This is formally specified as follows:

$$post(A) \wedge hyp \rightarrow pre(B)$$

where hyp is an hypothesis that, when combined with post(A), implies pre(B). Of course, the hypothesis hyp alone must not be sufficient to imply pre(B), that is we must have $hyp \rightarrow pre(B)$. Another requirement is that hyp must be consistent with post(A) because if this is not the case then one can derive anything from $post(A) \wedge hyp$, in particular pre(B).

The work we have done is based on this general definition of correlation. However, this first definition is not very manageable. Next sections present more practical definitions of correlation. The implementation of the correlation function in CRIM is actually based on these definitions.

4.2. Definition of alert correlation

Let A and B be two attacks and let Post(A) and Pre(B) respectively be the post condition of attack A and pre condition of attack B. Let us assume that Post(A) and Pre(B) respectively have the following form:

- Post(A) = $expr_{A1}$, $expr_{A2}$, … , $expr_{Am}$
- Pre(B) = $expr_{B1}$, $expr_{B2}$, … , $expr_{Bn}$

where each $expr_i$ must have one of the following forms:

- $expr_i$ = pred
- $expr_i$ = not(pred)
- $expr_i$ = knows(User,pred)
- $expr_i$ = knows(User, not(pred))

where pred is a predicate.

Definition 1: Direct correlation (simple case)

We say that attack A and attack B are directly correlated if the following condition is satisfied:

- there exists i in [1,m] and j in [1,n] such that $expr_{Ai}$ and $expr_{Bj}$ are unifiable through a most general unifier (mgu) θ.

For instance, attacks “MIR-0163” (NFS Mount) and “MIR-0164” (Modification of .rhost) are directly correlated. This is because post(“MIR-0163”) is equal to can_access(Source_user,Partition) and this predicate also appears in pre(“MIR-0164”). After renaming the variables of can_access(Source_user,Partition) that respectively appear in post(“MIR-0163”) and pre(“MIR-0164”) into can_access(Source_user1,Partition1) and can_access(Source_user2,Partition2), we can conclude that these expressions are unifiable through mgu θ such that Source_user1 = Source_user2 and Partition1 = Partition2.

On the other hand, the converse is not true, that is attack “MIR-0164” is not directly correlated with attack “MIR-0163”. This is because post(“MIR-0164”) is equal to access_level(Source_user,Target_address,user). Predicate access_level(Source_user,Target_address,remote) appears in pre(“MIR-0163”) but since constants *user* and *remote* are not unifiable, correlation of “MIR-0164” with “MIR-0163” fails.

Let us now try to correlate attack “MIR-0162” (Showmount) with attack “MIR-0163” (Mount partition). A possible post condition of “MIR-0162” is knows(Source_user,mounted_partition(Target_address,Partition)), that is the intruder Source_user knows what partitions are mounted on the target whose IP address is Target_address. On the other hand, mounted_partition(Target_address,Partition) appears in pre(“MIR-0163”). However, due to *knows* modality, this last expression is not directly unifiable with post(“MIR-0162”). This is intuitively not satisfactory since executing Showmount enables the intruder to then mount a partition observed in Showmount.

Therefore, we slightly modify definition 1 so that attack “MIR-0162” can be correlated with “MIR-0163”. This leads to the following definition:

Definition 2: Direct correlation (general case)

We say that attack A and attack B are directly correlated if one of the following conditions is satisfied:

- there exists i in [1,m] and j in [1,n] such that $expr_{Ai}$ and $expr_{Bj}$ are unifiable through a mgu θ.

or

- there exists i in [1,m] and j in [1,n] such that $expr_{Ai}$ and knows(User,$expr_{Bj}$) are unifiable through a mgu θ.

```
<?xml version=”1.0” encoding=”UTF-8”>
<rule ruleid=”RULE-0001”>

<pre> use_service(System_address,’NetBios’)
</pre>

<post> use_os(System_address,windows)
</post>

</rule>
```

Figure 4: Example of ontological rule

4.3. Indirect correlation

Let us now consider attacks “MIR-0073” (TCPScan) and “MIR-0036” (Winnuke”). These two attacks are not correlated using definition 2. However, attack Winnuke to succeed requires that the operating system used on the target system is Windows. The intruder can obtain this knowledge about the target system by performing TCPScan and by observing that port 139 is open (meaning that a NetBios session is open which is characteristic of Windows system).

Therefore, it would be suitable to correlate attacks “TCPScan” and “Winnuke” in the case where port 139 is scanned (and open). For this purpose, the solution we suggest is to specify ontological rules to represent possible relations between predicates. These ontological rules are also represented using a pre and post condition. Figure 4 shows an example of such a rule. This ontological rule says that if a system whose IP address is System_address uses service NetBios, then the operating system used on this system is Windows.

From a syntactical point of view, we assume that restrictions that apply to the representation of pre and post conditions in an ontological rule are similar to the one for pre and post conditions of an attack (that is, they do not include disjunction).

Next step is then to generalize definition 2 when ontological rule are used to perform correlation. This generalization is done in two steps. We first generalize definition 2 so that it applies to correlate two ontological rules or an attack with an ontological rule or an ontological rule with an attack. Since we assume that the syntactical format of the pre and post conditions of an attack is similar to the one of an ontological rule, this generalization is straightforward.

We then introduce the notion of indirect correlation. It is defined as follows:

Definition 3: Indirect correlation

We say that attack A and attack B are indirectly correlated through ontological rules R_1, …, R_n if the following conditions are satisfied:

- Attack A is directly correlated with rule R_1 through a most general unifier θ_0,
- For each j in [1,n-1], rule R_j is directly correlated with rule R_{j+1} through a most general unifier θ_j,
- Rule R_n is directly correlated with attack B through a most general unifier θ_n.

Using definition 3, we can now conclude that attack “MIR-0073” (TCPScan) is indirectly correlated with attack “MIR-0036” (Winnuke). This is because the post-condition of attack “MIR-0073” is equal to knows(Source_user,use_service(Target_address,Target_service)). Then, since the pre-condition of “RULE-0001” is equal to use_service(System_address,’NetBios’), “MIR-0073” is directly correlated to “RULE-0001” through the mgu:

- Target_address = System_address, Target_service = ‘NetBios’

Similarly, the post-condition of “RULE-0001” is equal to use_os(System_address,windows). Since this predicate also appears in the precondition of “MIR-0036”, “RULE-0001” is correlated with “MIR-0036”

when System_address = Target_address. Thus, we can conclude that attack “MIR-0073” is indirectly correlated with “MIR-0036”.

4.4. Generating correlation rules

In this section, we show how to automatically generate correlation rules similar to the one presented in section 3. The process we suggest is the following.

Let us consider two attacks Attack1 and Attack2 whose descriptions are correlated according to definition 2 through a mgu θ. After renaming the variables that appear in the descriptions of Attack1 and Attack2 so that there is no common variable in these descriptions, we shall generate a correlation rule having the following form:

correlation_rule(Alert1,Alert2) :-
  cond_detection(Attack1),
  cond_detection(Attack2),
  θ.

where Alert1 and Alert2 are respectively the (renamed) variables that appear in the detection field of Attack1 and Attack2[3]. For example, figure 5.a presents the correlation rule corresponding to attacks “MIR-0163” (NFS Mount) and “MIR-0164” (Modification of .rhost).

This rule is correct but it is not fully optimized[4]. In particular, target descriptions of the two alerts might be removed since they are not related to the correlation condition. Notice also that our process also generates condition Partition1 = Partition2. This is correct since, in this attack scenario, the intruder must modify the .rhost file of a partition previously mounted with attack NFS Mount. But, as Partition1 and Partition2 remains free variables, this condition will be always evaluated to true. This is because we assume that information about the mounted partition is not provided by alerts corresponding to NFS Mount and Modification of .rhost.

The case where two attacks Attack1 and Attack2 are indirectly correlated using ontological rules is slightly more complicated. If Attack1 and Attack2 are indirectly correlated using ontological rules R_1, …, R_n through a set of mgu θ_0, …, θ_n, then we shall generate a correlation rule having the following form:

correlation_rule(Alert1,Alert2) :-
  cond_detection(Attack1),
  cond_detection(Attack2),
  θ_0, …, θ_n.

For example, figure 5.b presents the correlation rule corresponding to attacks “MIR-0073” (TCPScan) and “MIR-0036” (Winnuke).

Notice that all the correlation rules are automatically generated by analyzing the descriptions in LAMBDA of the set of attacks. This process is performed offline and therefore, it is not time consuming for online intrusion detection.

4.5. Applying correlation rules

After all the correlation rules are generated offline, the online correlation process can start. When this process receives a new alert Alert1, it proceeds as follows.

Let Attack1 be the classification associated with Alert1. In a first step, we check if there are other alerts already stored in the database and whose classification is Attack2 such that the fact attack_correlation(Attack1,Attack2) or attack_correlation(Attack2,Attack1) is stored in the correlation base. Notice that this first step is only for optimization since the correlation rules might be applied directly.

---

[3] Notice that our approach does not apply to the case where the detection field of attacks corresponds to combined events. It is restricted to detection scenarios that are single events.

[4] Defining an algorithm to optimize correlation rules might be done. But, the overhead this seems to cause on performance is marginal so that we do not find that such an optimization is a priority.

```
alert_correlation(Alert1,Alert2) :-

 alert(Alert1),
 source(Alert1,Source1),
 source_user(Source1,Source_user1),
 target(Alert1,Target1),
 target_node(Target1,Target_node1),
 address(Target_node1,Target_address1),
 classification(Alert1,"MIR-0163"),

 alert(Alert2),
 source(Alert2,Source2),
 source_user(Source2,Source_user2),
 target(Alert2,Target2),
 target_node(Target2,Target_node2),
 address(Target_node2,Target_address2),
 target_user(Target2,Target_user2),
 classification(Alert2,"MIR-0164"),

 Source_user1 = Source_user2,
 Partition1 = Partition2.
```

Figure 5.a: Correlation rule for "MIR-0163" (NFS Mount) and "MIR-0164" (Modification of .rhost)

```
alert_correlation(Alert1,Alert2) :-

 alert(Alert1),
 source(Alert1,Source1),
 source_node(Source1,Source_node1),
 address(Source_node1,Source_address1),
 source_user(Source1,Source_user1),
 target(Alert1,Target1),
 target_node(Target1,Target_node1),
 address(Target_node1,Target_address1),
 target_service(Target1,Target_service1),
 classification(Alert1,"MIR-0073"),

 alert(Alert2),
 source(Alert2,Source2),
 source_node(Source2,Source_node2),
 address(Source_node2,Source_address2),
 target(Alert2,Target2),
 target_node(Target2,Target_node2),
 address(Target_node2,Target_address2),
 classification(Alert2,"MIR-0036"),

 Target_address1 = System_address3,
 Target_service1 = 'NetBios',

 System_address3 = Target_address2.
```

Figure 5.b: Correlation rule for "MIR-0073" (TCPScan) and "MIR-0036" (Winnuke)

Figure 5: Examples of correlation rules

However it is more efficient to first filter on predicate attack_correlation to check if there are alerts that are potentially correlated to Alert1. Notice that we both look for facts attack_correlation(Attack1,Attack2) and attack_correlation(Attack2,Attack1) because we do not assume that the alerts are received in the same order as their order of occurrence.

If there are alerts Alert2 that are potentially correlated with Alert1, then the corresponding correlation rules apply to check if the correlation conditions are satisfied. The result is a set of pairs of alerts that are correlated, one member of these pairs being Alert1. For each pair in this set, we then apply an algorithm to check if this pair might be aggregated to an existing scenario. Else, a new scenario starting with this pair of alerts is generated.

For instance, let us assume that there is already a scenario with three alerts (alert1, alert2, alert3). Let us assume that alert4 is received and that the online correlation process generates a pair (alert3, alert4). In this case, a "longer" scenario (alert1, alert2, alert3, alert4) is generated.

Notice that a complex scenario with several branches is actually decomposed into several sequential scenarios corresponding to each branch. For instance, let us consider the scenario presented in figure 6. It is represented by two sequential scenarios (alert1, alert2, alert3, alert4) and (alert2,alert5,alert6,alert4). It will be the role of the graphic interface to "aggregate" these two sequential scenarios as presented in figure 6 (see annex 1 for a short presentation of this interface).

For each sequential scenario, the online correlation process generates a special alert called "scenario alert". This alert is fully compliant with the IDMEF format. The "Correlationalert" field of this alert corresponds to the list of correlated alerts (the order in this list is important!). The other fields of this alert are generated by using the merging function to merge data contained in the correlated alerts (see [Cup01] for more details about the merging function).

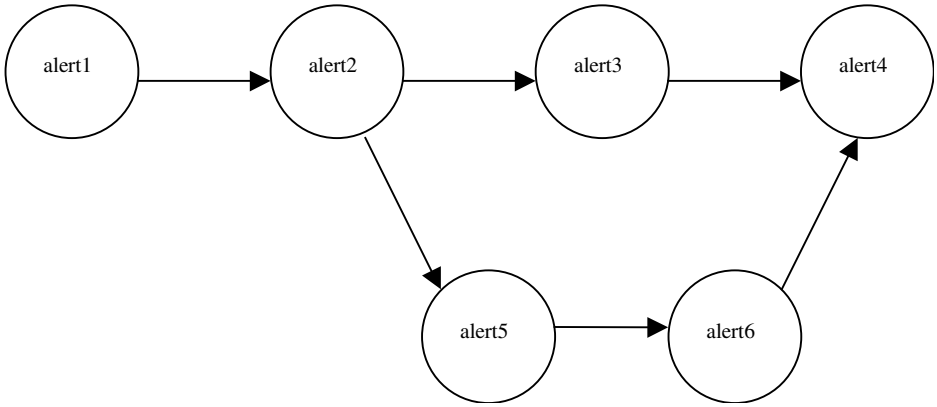


Figure 6: Example of attack scenario

5. Abductive correlation

There are still some problems that arise when we apply our online correlation process. For instance, let us consider the attack scenario presented in figure 7. We call this attack "illegal nfs mount". This attack scenario enables the intruder to get a root access by exploiting a misconfiguration in the security policy, namely the intruder can mount a partition corresponding to the home directory of root. There are 6 different steps in this attack.

These 6 steps are specified in Lambda so that when we apply the offline correlation process, we obtain 6 correlation rules as shown in Figure 8. cond1, … cond6 correspond to the 6 correlation conditions associated with these correlation rules.

The result provided by the offline correlation process should enable the online correlation process to fully recognize the "illegal NFS mount" scenario. However, when this attack is launched on a system supervised by Snort and e-Trust, 9 alerts are generated: 7 by Snort and 2 by e-Trust. Our clustering function gives 5 clusters. Actually, both Snort and e-Trust did not detect step 5.

Intrusion scenario	**Detection results**	**Fusion results**
Step 1 : attack_finger **finger** root	Snort : **3** alerts eTrust : 0 alert	CRIM : **1** alert
Step 2 : attack_rpcinfo **rpcinfo** <target>	Snort : **1** alert eTrust : **1** alert	CRIM : **1** alert
Step 3 : attack_showmount **showmount** <target>	Snort : **1** alert eTrust : 0 alert	CRIM : **1** alert
Step 4 : attack_mount **mount** directory	Snort : **1** alert eTrust : 0 alert	CRIM : **1** alert
Step 5 : attack_rhost cat "++" > .**rhost**	**Not detected**	
Step 6 : attack_rlogin **rlogin** <target>	Snort : **1** alert eTrust : **1** alert	CRIM : **1** alert

Figure 7: "Illegal NFS Mount" scenario

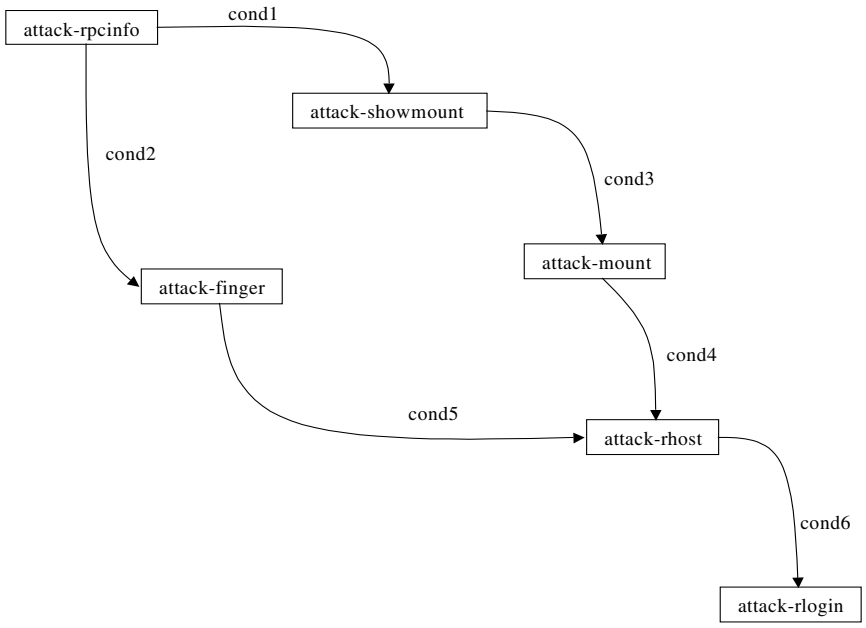


Figure 8: results of correlation process on attack "illegal NFS mount"

This result is then provided to the correlation function for further analysis, the objective being to correlate these 5 clusters in order to recognize one single complex attack. The main difficulty to recognize multiple steps attack "illegal NFS mount" comes from the fact that step 5 ("MIR-0164": attack_rhost) is not detected by both Snort and e-Trust. Our approach to solve this problem is the following.

The correlation function receives one alert corresponding to attack "MIR-0163" (attack_mount) and another one corresponding to "MIR-0165" (attack_rlogin). Since the correlation function knows that it is possible to correlate attack_mount with attack_rhost and then attack_rhost with attack_rlogin, the correlation function makes the hypothesis that it is possible to *transitively* correlate attack_mount with attack_rlogin[5]. In this case, the approach is to abduce[6] a new alert that is to create a new (virtual) alert corresponding to attack_rhost.

When this virtual alert is generated, its correlation field is initialized to "MIR-0164" (corresponding to attack_rhost). All the other fields are initialized by using Skolem functions. For instance, the target field is initialized to "target(1)" meaning that the target of this alert exists but is currently unknown. The case of detect time field is slightly more complex. It is partly unknown but must satisfy the following constraint: it must be after the alert corresponding to attack_mount and before the alert corresponding to attack_rlogin. The solution in this case is to manage interval of time. Due to space limitation, this point is not developed here but we plan to present it in a forthcoming paper.

Once the virtual alert is abduced, the online correlation process applies the corresponding correlation rules. It first checks if cond4 is satisfied. In particular, we have to satisfy the following condition: Target_address1 = Target_address2 where Target_address1 is the IP address of alert corresponding to attack_mount and Target_address2 is the IP address of the virtual alert. Since Target_address2 is actually a Skolem function, this unification succeeds and Target_address2 is updated so that it is now equal to Target_address1. Then, we have to check if cond6 is satisfied. cond6 includes a condition having the form Target_address2 = Target_address3 where Target_address2 is the IP address of the virtual alert and Target_address3 is the IP address of alert corresponding to attack_rlogin. But Target_address2 is now equal to Target_address1, so that checking cond6 will only succeed if Target_address1 is actually equal to Target_address3, that is the alerts corresponding to attack_mount and attack_rlogin have a target with the same IP address. Else cond6 is not satisfied and therefore the abductive correlation does not succeed.

We shall proceed similarly for cond5 to check whether it is possible to correlate the alert corresponding to attack_finger with the virtual alert. In this case, we have also to satisfy the following condition:

---

[5] Similar reasoning applies to transitively correlate attack_finger with attack_rlogin.

[6] Abduction consists is making new hypotheses and to use them to derive new facts. Typically, this kind of reasoning is used when facts are missing to complete a diagnostic.

Target_address4 = Target_address2 where Target_address4 is the IP address of alert corresponding to attack_finger and Target_address2 is the IP address of the virtual alert. This condition is satisfied if the alerts corresponding to attack_mount and attack_finger have a target with the same IP address.

In our experiment, abduction of an alert corresponding to attack_rhost succeeds. Therefore, the online correlation function correlates all the alerts generated by the attack "illegal nfs mount" into one single scenario, even though step 5 of this attack is not detected (see the appendix for a graphical presentation of the corresponding detection).

6. Conclusion

In this paper, we have presented the approach we suggest to designing the correlation function of CRIM, a cooperative module for intrusion detection systems. After specifying an attack base in Lambda, the offline correlation process analyzes these attack descriptions to automatically generate a set of correlation rules. The online correlation process then applies these correlation rules on the alerts generated by the IDS to recognize more global attack scenarios.

This approach has been implemented in GNU-Prolog [Dia00] with a graphic interface in Java (see the appendix for a view of this graphical interface).

It is important to observe that alert correlation is very useful to reduce the number of false positives. For instance, notice that separately each step of the intrusion scenario "illegal NFS Mount" presented in section 5 might actually correspond to a false positive. It is only after correlating alerts that we can derive that an intrusion occurred. Therefore, in many cases, it is possible to conclude that an alert that is not further correlated with other alerts is actually a false positive.

There are several issues to this work.

First, we are currently designing the intention recognition function. The main objective of this function is to anticipate on how the intruder will go on. To achieve this objective, we are actually combining two approaches:

- Abductive correlation is used to forecast next step of the attack scenario. This first approach is based on the analysis of facts attack_correlation(Attack1,Attack2) and is therefore simpler than the abductive correlation process presented in section 5 since virtual alerts are not generated in this case.
- Specification of global intrusion objectives. This idea is quite similar to specifying a security policy since we may assume that the intruder's objective is to violate the security policy (at least from a defensive point of view!). A global intrusion objective might be viewed as a logical expression describing the target's state the intruder wants to achieve. The principal of this second approach is then to correlate attacks with intrusion objectives(corresponding to the violation of the security policy). This would guide the intention recognition process.

Of course, in both cases, the intention recognition function may provide several possibilities, that is several "next steps" in the first approach and several possible intrusion objectives in the second approach. Therefore, we have also to design an approach to choose the "best" possibility. This is still an open research problem.

This briefly sketch our approach for the intention recognition function. We plan to provide more details about this function in a forthcoming paper.

Second, we plan to encode a larger base of attacks in LAMBDA. The objective here is to improve the correlation results but also to check whether it is possible to use our correlation approach to discover new attack scenarios by searching how to correlate elementary attacks.

Finally, notice that the architecture we suggest in this paper is centralized, the CRIM module receiving all the alerts generated by the IDS. This is mainly due to technical constraints since we consider in the MIRADOR project that it was not practical to directly create communication between IDS. Such a distributed approach was suggested in [YN+00]. The authors illustrate their approach with the Mitnick attack. There are two steps in this attack. In the first step, the intruder floods a given host H. Then the intruder sends spoofed messages corresponding to H address to establish a communication with a given server S. When S sends an acknowledge to H, H cannot ask to close the connection since it is flooded. In a distributed approach, a first IDS can detect that H is flooded and then asks a second IDS to detect whether S continues receiving messages with H address. If this is the case, then we can conclude that the Mitnick attack is occurring. We agree that this distributed approach is interesting and we plan to analyze it in the future.

Acknowledgements

This work was funded by the DGA/CELAR/CASSI as a part of the Mirador project. The author would also like to thank all the members of this project: Jacques Capoulade, Mamadou Diop, Samuel Dubus, Aldric Feuillebois (Alcatel CIT), Patrice Carle (ONERA), Ewan Cochevelou, Sylvain Gombault (ENST-Bretagne), Laurent Heye, Ludovic Mé and Cédric Michel (SupElec Rennes).

References

[CA00] Computer Associates. E-Trust Intrusion Detection. 2000.

[CD01] D. Curry and H. Debar. Intrusion Detection Message Exchange Format Data Model and Extensible Markup Language (XML) Document Type Definition. draft-itetf-idwg-idmef-xml-03.txt, February 2001.

[Cup01] F. Cuppens. Managing alerts in a multi-intrusion detection environment. 17th Annual Computer Security Applications Conference (ACSAC). New-Orleans, December 2001.

[CO00] F. Cuppens and R. Ortalo. LAMBDA: A Language to Model a Database for Detection of Attacks. Proceedings of the Third International Workshop on the Recent Advances in Intrusion Detection (RAID'2000), Toulouse, France, October 2000.

[Dia00] M. Diaz. GNU Prolog: A Native Prolog Compiler with Constraint Solving over Finite Domains. Edition 1.4 for GNU Prolog version 1.2.1. http://gnu-prolog.inria.fr/manual/. July, 2000.

[DW01] H. Debar and A. Wespi. The Intrusion-Detection Console Correlation Mechanism. Workshop on the Recent Advances in Intrusion Detection (RAID'2001), Davis, USA, October 2001.

[Hua98] M.-Y. Huang. A Large-scale Distributed Intrusion Detection Framework Based on Attack Strategy Analysis. Proceedings of the First International Workshop on the Recent Advances in Intrusion Detection (RAID'98), Louvain-La-Neuve, Belgium, 1998.

[Ken99] K. Kendall. A Database of Computer Attacks for the Evaluation of Intrusion Detection Systems. June 1999.

[Lee99] W. Lee. Combining Knowledge Discovery and Knowledge Engineering to Build IDSs. Proceedings of the Second International Workshop on the Recent Advances in Intrusion Detection (RAID'99), Purdue, USA, October 1999.

[Lip99] R. Lippmann. Using Key String and Neural Networks to Reduce False Alarms and Detect New Attacks with Sniffer-Based Intrusion Detection Systems. Proceedings of the Second International Workshop on the Recent Advances in Intrusion Detection (RAID'99), Purdue, USA, October 1999.

[MM01] C. Michel and L. Mé. Adele: an Attack Description Language for Knowledge-based Intrusion Detection. *16th International Conference on Information Security*. Kluwer. June 2001.

[Roe99] M. Roesch. Snort – Lightweight Intrusion Detection for Networks. Proceedings of USENIX LISA'99, November 1999.

[VS01] A. Valdes and K. Skinner. Probabilistic Alert Correlation. Proceedings of the Fourth International Workshop on the Recent Advances in Intrusion Detection (RAID'2001), Davis, USA, October 2001.

[YN+00] J. Yang, P. Ning, X. Wang, and S. Jajodia. CARDS: A Distributed System for Detecting Coordinated Attacks. IFIP TC11 Sixteenth Annual Working Conference on Information Security, August 2000.

[Zer99] D. Zerkle. A Data-Mining Analysis of RTID. Proceedings of the Second International Workshop on the Recent Advances in Intrusion Detection (RAID'99), Purdue, USA, October 1999.

Annex 1: Graphical interface

Figure 9 presents a view of CRIM interface. There are 3 sub-windows in this interface. The upper window provides a view of alerts that are directly generated by the IDS connected to CRIM or abduced by CRIM (virtual alerts). The central window corresponds to fusion alerts, that is alerts generated by the merging function of CRIM. Finally, the lower window presents the alerts generated by the online correlation function. It actually shows detection of the “Illegal NFS Mount” scenario presented in Section 5. Alertid_1 is a virtual alert corresponding to attack “MIR-0164” (Modification of .rhost) that is automatically abduced to complete detection of this scenario.

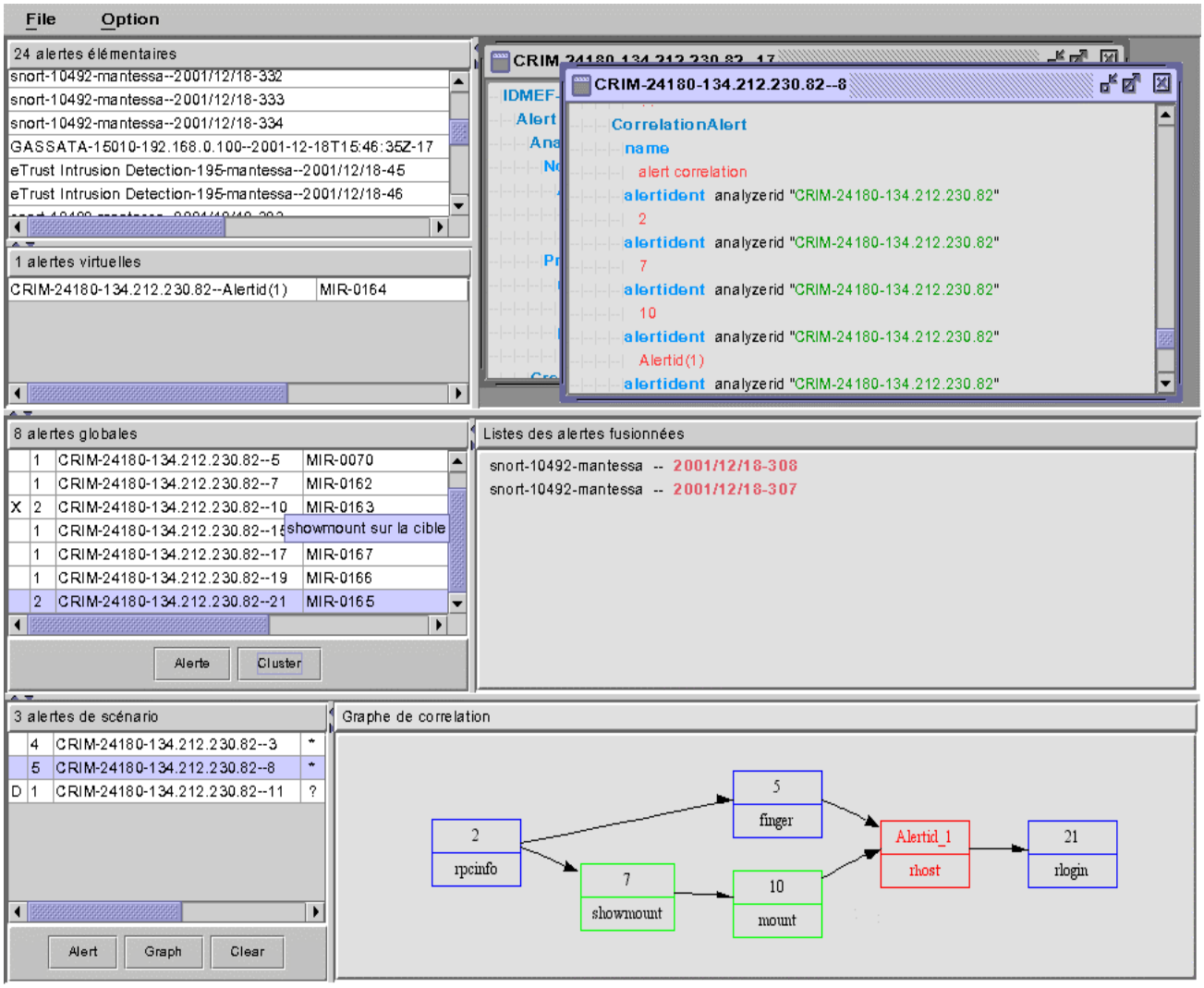


Figure 9 : Graphical Interface

Click here to view PowerPoint presentation; Press Esc to exit

Real Time Intrusion Detection - Applying Correlation and Fusion for Outside the Network Attack Forecasting and Insider Attack Detection

Mr. Dennis McCallam
Northrop Grumman
Information Technology
1831 Weihle Ave.
Reston, VA 20190
United States

Mr. John Whitson
Northrop Grumman
Information Technology
MS 801-026
South Oyster Bay Rd.
Bethpage, NY 11714
United States

Mr. Paul Zavidniak
Northrop Grumman
Information Technology
4010 Sorrento Valley Blvd.
San Diego, CA 92121
United States

ABSTRACT

As a result of the rapid growth in computer technology, the government and private sector, has become extremely dependent on automated information systems. Access to these systems may be wanted by individuals or organizations seeking monetary gain, political blackmail or those dedicated to causing damage. As a consequence, access to systems is restricted and controlled to those who are trusted and approved. Given that, the issue of attack detection has fallen into two categories: insider and outsider. The majority of research and implementation to date has been to consider both cases independently, but in fact they are related through the axiom that "When an outsider penetrates the perimeter, they are now an insider with access." We have looked carefully at this issue and have been applying techniques of correlation and fusion to successfully address the outside-inside attack problem. The techniques are the same, but the events and data sources used for fusion and correlation span both the cyber and physical.

Looking at the insider attack, we have found that integrating and fusing information yields a strong indicator that something is not as it seems and then allows real time intervention as a course of action. The point of the integration is to fuse together pieces of information that will provide a real-time picture of what is going on. Using user id/password, physical access, IP net address, and the like provides increased accuracy in the declaration of an insider attack while reducing the false alarm potential of using fewer pieces of information. This offers a great deal of promise not only in the development of a unique approach for insider detection, but also in the operational transition primarily because the additional filtering components are commercially available and are accepted. This can be coupled with other operational choices as to: closing off the system to the intruder, gathering forensic evidence on the intruder, and/or employing tactical deception against the intruder.

Correlation is one of the most important discriminators when assessing the impact of large-scale attacks, whether attacked using a virus-based manifestation or a traditional paradigm. The Network Early Warning System (NEWS) is a prototype system that addresses these high-impact attacks at multiple levels, and will be ready for deployment in the near future. NEWS departs from traditional security systems by standing away from individual IDS (so as not to impact their function), yet interfacing with them to

Paper presented at the RTO IST Symposium on "Real Time Intrusion Detection", held in Estoril, Portugal from 27-28 May 2002, and published in RTO-MP-101.

provide an advanced multi-IDS correlation/association/fusion capability towards cyber-situational assessment, as well as a forecasting and course of action function for on-the-horizon cyber-threat mitigation. This paper highlights some of the key techniques used successfully within NEWS to achieve these goals, gives specific direction on pitfalls that are often encountered in the practical deployment of such systems, and describes an approach to early warning of network worm/virus infestations.

The combination of fusion and correlation lead us to believe that this will improve information warfare capabilities by offering an additional level of real-time intrusion detection that is performance based.

Bounding the Problem

Before discussing a solution path, it is important to look at the characteristics of the problem area and develop a set of boundary conditions. The process sought to examine the differences and similarities between outsider and insider attacks and then apply those observations towards the solution. The ensuing discussion looks at both the outside the system perimeter attacker and the insider the enclave attacker. Consideration went to both methods and motives.

The outsider is constantly probing at the system defenses. For the most part, they use readily available scripts with the more experienced attacker developing new scripts. The scripts exploit publicized vulnerabilities found at a number of websites. In executing the attack, there are sets of steps that the attacker follows. In fact, there is an ensuing section in this paper on the information warfare timeline that illustrates those steps. Since the attacker needs to follow an overall plan to successfully gain entry to the system, those steps are ordered and predictable. For the defender, there is some good news and that is that the attacker is more likely to cause a defensive alert and therefore be detected at some point. The insider already has some knowledge about the system and does not have to go through the process of gaining general access. This knowledge allows the insider to avoid the perimeter defenses the outsider needs to contend with. More often, the insider leverages accesses and privileges and uses that to mask their true identity.

There are many studies that have been done profiling the attackers and looking at personality traits, habits, etc. The concern here dealt more with purpose. A brief look at motives was illustrative. A great many of the outside attacks seek to just gain entry, perform a little mischief (such as a web site modification) and then leave. The point of this type of attack is more a right of passage rather than anything else. Some external attacks seek to overload the defenses of the system and prevent responses. The ones of concern are those that are dedicated to causing malicious activity after system access has been obtained. And this is where there is commonality between insider and outsider motives. The outsider at this point has, in effect, become an insider. They already have access to the system and in some cases have gotten root access and altered privileges. An insider already has that access. The next step took a further look at the motives and results of previous attacks. The observation was that the purpose in gaining access to a system was usually premeditated and would seek to cause some form of system damage. Motives for causing the damage range from disgruntled employees to terminated network managers. One specific damage, as in the Omega Engineering case, was designed to

cripple the business[1]. Another concern includes the stealing of secrets or information contained in the system for various forms of business and military espionage. A form of tactical deception would alter certain pieces of data within the system to cause subtle computational errors that might not be noticed for a length of time, if at all as in the Citibank case[2].

Conducting investigations into attacks has been more of a post-mortem analysis conducted after the damage has been done. Post mortem solutions have a fundamental weakness in that the analysis is subject to only looking at the information and data residuals after the incident has occurred, and many times long after the incident. But more fundamental is the fact that the intruder could have left only the information that he/she wanted us to have. During the attack, the data could be manipulated in such a way that an analysis of only what remains could lead to erroneous conclusions.

The Temporal, Cyber, and Physical Relationships

Temporal structure and temporal analysis is a theory put forth by Victoria Koehler-Jones[3]. The two important elements of the temporal structure are in the pattern (or the way time and events move from the past, through the present and into the future) and the flow of those events. The point of the temporal analysis is to analyze and then to provide a quantitative assessment of the importance of the events. The postulation is that each person has a temporal signature or temporal identity. And just as important, the way those events are viewed can cause differences in the inferences.

Events to be considered in the fusion process were both cyber and physical with both having a temporal element. For each event that occurs, there is a time associated with it. In reconstructing actions and in the fusion process, the time becomes an important component. When events are correlated and combined, certain dependencies can be either established or checked for. For example, to log on at a workstation inside the compound, a user must have entered the compound.

The next step considered cyber events and physical events and provided some descriptors and definitions of each. Cyber events are considered to be those events involved a computer system and the information contained within that system. Cyber events began

---

[1] On May 9, 2000, Timothy Lloyd was convicted of writing six lines of code essentially, a code "bomb" that obliterated Omega Engineering Corporation's design and production programs. Since Omega makes components for clients such as NASA and the US Navy, those systems were the company's rainmakers. Lloyd worked at Omega for for 11 years. Three weeks after Lloyd was fired, a worker at Omega's manufacturing plant in Bridgeport, New Jersey, logged on to a computer terminal. It was July 31, 1996, the date that the bomb was set to detonate. By logging in, the worker unleashed the aberrant code that instructed the system to delete the software running Omega's manufacturing operations.

[2] In 1994, Russian hacker Vladimir Levin engineered a heist from Citibank the hacker community's first foray into big-money banking tricking the company's computers into distributing an estimated $10 million to him and his accomplices in several countries. Levin used passwords and codes stolen from Citibank customers to make transfers to his accounts.

[3] From “Disaster Responders Perception of Time”, Victoria Koehler-Jones, University of Nevada Las Vegas http://www.library.ca.gov/CRB/96/05/over_8.html

with access to a computer system by subverting authentication or providing alternate forms of identification that the system accepted. In many cases, insiders did not want to leave evidence directly linking them to an attack. The intent of the cyber event was information. But in each case, there is an operation the attacker uses that could cause an alert from one of the defenses. These events span a wide range and could be as basic as incorrect password/user id attempts at log on, continual attempts to access protected or limited access sections of the system. There are actions that the attacker can take to include attempts to install unauthorized software or components, reconfiguration of system asset attempts, and placement of back doors.

There are a number of devices that provide and regulate access to areas, buildings, and rooms. All of these record the access attempt and any time associated with that attempt. The physical events are those that involve a specific and defined geo-physical location. Particularly, this includes the point of exact physical coordinate placement within a building (such as floor and room number) to an exact spatial coordinate for mobile units. For example, a cyber event could be the act of logging on to a system while the corresponding physical portion of that event is the location of the workstation on the network. If there is static IP addressing used, for example, an exact location on the network could be defined. Similarly this works for building entry and within a building, specific area entry. All of these can be defined spatially and all have an associated time. It becomes significant in the fusion process when relationships can be drawn between the cyber and physical events. For example, the cyber event of logging on to a system can not occur if the person logging on has not been recorded as entering the building. There is either a security violation taken place on the entry (the common practice of 'tailgating' through the badge reader), or there is in fact an attempt to use another identity to gain access. In either event, the connection to the system can be terminated and other appropriate actions can be undertaken.

The Information Warfare Timeline Concept[4]

If one could take a God's eye view of an information warfare attack, and really examine it in detail, two key points become evident. The first is that there is a precedence and relationship to many of the events that occur in an attack. For example, before a system can be effectively attacked, attack access points need to be established by the attacker. Before the attack entry points can be developed, the attacker must gain access to the system. To successfully gain access, the attacker must avoid being detected by the system firewalls and defenses. To gain that access, the attacker must perform surveillance and understand the nuances of the system to be attacked.

Therefore, we can identify and examine generic events and the anticipated sequence of those events of an IW attack. We can then understand what events are attacker-dependent and what reactions are defender-dependent. To begin with, we can construct a visual, or timeline, representation sequencing generic information warfare events. Figure 1 shows a time-sequenced overview of the generic actions that will happen given

[4] Early on, we at Northrop Grumman developed the IW Timeline concept as a way of explaining the events leading up to and following a cyber attack. This discussion has been reproduced in various forms and remains after 6 plus years a good basis for understanding attacking and defending postures.

an intrusive IW attack. (We understand that other categories of attack, such as virus attacks or spamming may not follow this timeline.)

There are a number of things that need to happen prior to an IW attack and a number of things that must occur after an attack. Each of the events we depict on the IW Timeline in Figure 1 are briefly described below.

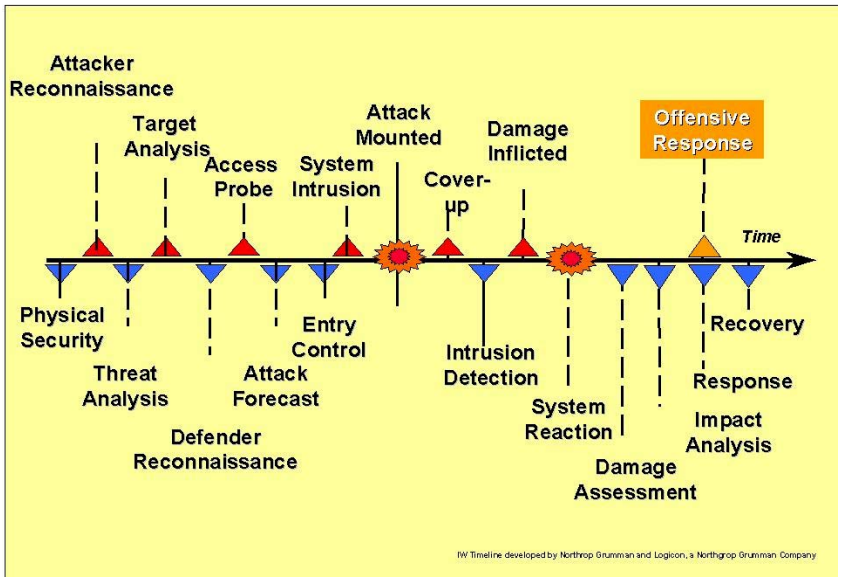


Figure 1
The Information Warfare Timeline

- Physical Security refers to the security devices that restrict physical access to computing systems. These include "dumb" devices such as locks, as well as "smart" devices such as facial recognition systems and fingerprint access control (e.g. TrueTouch's Biometric Software Security Suite, IriScan's Iris Recognition Technology).
- Attacker Reconnaissance refers to those actions taken by the attacker to "scout out" a system electronically and through other forms of research. For example, a ping could be done to the system that leads to the mapping of the network and identification of computing and network resources that comprise the system. This can then be augmented with widely available information on vulnerabilities of the operating system, the router, etc. all of which forms a knowledge base for planning the access portion of the attack.
- Defender Reconnaissance refers to those actions taken by a defender to "scout out" potential attackers and their potential entry points. Some enterprises gather data on potential attackers via "honey pot" web sites designed to lure people interested in attacking entities associated with specific issues. For example, a company or agency engaged in controversial business (e.g. tobacco companies, intelligence agencies, major oil companies during a fuel shortage, political groups) may set up sites with provocative content designed to attract attackers, much like bees to a honey pot. When an unsuspecting person visits a honey pot site, information is taken about the visitor without his knowledge. Such defensive surveillance can be augmented at later

points in the timeline as information collected on potential attackers is combined with other anomalous behaviors and observed patterns (i.e., as in a pattern of origin). Information Protection Managers can also use commercial vulnerability scanners (e.g. Internet Security System (ISS)'s Internet Scanner, Netect's HackerShield,) to conduct reconnaissance on their own systems, to determine the weak points that could be exploited by an attacker.

- Target Analysis refers to those actions taken by the attacker that augments the results of the reconnaissance with information available to the attacker on the Internet or through other sources. For example, the reconnaissance could show that the target system is built from Dell boxes, running Windows NT, and using a Novell network. The attacker then can visit several hacker web sites to get known vulnerabilities of these components.
- Threat Analysis refers to those actions taken on the defensive side to further identify potential attackers and their motives. For example, some of the honey pot sites referenced above were found to be military intelligence sites for foreign governments. The "honey" in those cases was supposed to be descriptions of foreign military equipment.
- Access Probe refers to those actions taken that probe the system for access beginning at the fundamental user id/logon all the way to full root access.
- Forecast refers to integration of information by the defenders that allow forecasting of attacks to be made. Typically, this takes the form of information from the reconnaissance and surveillance phases analyzed and synthesized with neural nets and related technologies (e.g. Northrop Grumman's Network Early Warning System).
- Entry Control refers to those measures that are taken to electronically restrict access to a system. These controls may be multiple passwords, card readers integrated into the system control, firewalls, etc. (e.g. Check Point's FireWall-1 4.0, Cisco's PIX Firewall). Entry control devices are also those pieces of software that reside in the router or other portions of the network that automatically allow or deny access.
- System Intrusion refers to the event of actually gaining the access into the system.
- Attack Mounted refers to the actual attack itself, which may occur as a single event or as a wave of attack events.
- Intrusion Detection refers to the synthesis of information that allows the system to realize that an intrusion has either been completed or is in the process. This is frequently accomplished with commercially available host- or network-based intrusion detection systems (e.g. ISS's RealSecure, Axent's Intruder Alert, or Network Associates, Inc. (NAI)'s CyberCop). The "in-the-process" portion is sometimes referred to as "indications and warnings" where detection is accomplished using reconnaissance and/or access probe information. This is one of those areas on the timeline where the actual knowledge of an intrusion can occur during a period of time that can stretch all the way from the reconnaissance through an attack itself. Obviously, the more sophisticated the detection process, the earlier in the timeline an intrusion can be either detected or predicted
- Cover up refers to those actions taken by the intruder to cover up and eliminate evidence of the intrusion and the attack.
- Damage Inflicted - Refers to the results of the attack in terms of corruption or loss of information and/or system functionality. In any event, this also defines a level of trustworthiness or resiliency that the system still has.

- System Reaction refers to the changes in system operation which result from the attack (e.g. every other blip on the air traffic controller's screen disappears, or the wrong people get billed for overseas phone calls). These reactions could be designed into the attack itself or occur as the result of the system going into a different mode and then failing because of compromised information directly related to that mode.
- Damage Assessment refers to the assessment of the functionality and/or information loss of the system (i.e. electronic battle damage assessment).
- Impact Analysis refers to the defender's projection of the impact of the attack on the business process or mission operations. Ideally, "What if?" impact analyses would be done long before any attack occurs. In that case, procedures and look-up tables would be created to advise the Information Protection Manager about the seriousness of the impact of various attacks on standard operations.
- Response refers to the actions taken by the Information Protection Manager to respond to an attack. Examples include: shutting down parts of the network, eliminating services such as FTP or e-mail from certain network nodes, initiating evidence collection, etc.
- Recovery refers to those actions that are taken to reconstitute the information in a system and return the system to full operational and trusted conditions. The optimum goal of recovery is to non-intrusively restore the system in real-time with no manual interaction.

Issues in Forecasting and Fusion

" It is of the highest importance in the art of detection to be able to recognize out of a number of facts which are incidental and which vital. Otherwise your energy and attention must be dissipated instead of being concentrated." - Sherlock Holmes[5], The Reigate.

The main postulation of this research activity was that fusion of information would yield better results in detecting an insider attack and is able to do so in real time. Most security around computer networks has forms of physical security, such as locks, electronic entry devices, etc. These are monitored by security personnel who look only at the entry/exit logs with any correlation done manually and well after any fact. For example, if something is stolen from a closed area, the logs can be examined to see who may have had opportunity and was present when the event occurred. Equally as important, the logs can be examined to see who wasn't present, and they can be ruled out as suspects. This example taken with several sources of information is exactly the context of data fusion as it relates to this research. Data fusion looks at information garnered from several sources and then draws inferences and conclusions from that set. The expectation is that by combining various inputs, the accuracy of those inferences would increase over results achieved from using individual inputs. The two questions we are trying to answer are:

- Are there authentication inconsistencies? and,

[5] From the website of Sherlock Holmes quotes, www.bakerstreet221b.de/canon/index.html. There are a number of quotes across the works of Sir Arthur Conan Doyle that are applicable to the concept of fusion. In fact, Sherlock Holmes was probably the first literary detective to demonstrate the art of data fusion as applied to solving crimes.

- How can we substantiate?

The first type of correlation looks at the authentication consistencies and the authentication inconsistencies to ascertain that a given person is in fact, who they claim to be and. Consider the case of a valid user id/password combination, a token for access into other parts of the system, devices (such as card readers) that allow physical access to computer equipment and information from the timekeeping system. Suppose the following case is considered: a Joe ID - Joe Password are valid users of the system, Joe Access Card, and Joe's record of his time for this week. One form of consistency is that Joe used his access card to gain entry to the area, then he used his Joe ID and Password to log on to the system the system, and Joe has time entered on his timecard for work on Project X. Taken one at a time, each of the data items are valid and would indicate that the person logged onto the system is in fact, Joe. This is an authentication consistency. But suppose the time card indicates Joe is on vacation. Again, each of the individual data items are valid and if taken one at a time, would indicate Joe could be on the system. But taken as a unit, they show an authentication inconsistency in Joe actually being on-site. At this point, there are courses of action that can be taken, the least of which would be to terminate the access from 'Joe' until a follow-up can be completed. In essence, we are developing a process where we are trying to prove that someone logged in to a computer is either who they clam to be or is impersonating a legitimate user, we have a good fit. In fact what we are trying to accomplish is to view the data, correlate where appropriate and present evidence.

All this led to five primary observations on fusion:

- There is correlation between the timeline events that allows time synchronized and normalized event data to be mapped into the various stages of an attack. This allows some expectation as to next event, by looking at the timeline and being alert for subsequent operations.
- Associated events are essentially different actions undertaken by and are attributed to the same user. These could include for example the entry to the room, logon, logoff, and exit. When these events are associated together, they describe the movements of the user. There are also like events that are the same event, but different user. The bottom line correlates associated events and then discriminates like events.
- Key to the fusion process is the formulation of beliefs as to what will or may happen next. The foundation of the belief rests on what partial evidence has been observed and seen.
- Once a foundation of beliefs has been established, information garnered from other sensors can check the hypotheses. This can occur by tasking other sensors to provide information that can be used to further the verification, validation, and refinement of the belief.
- Finally, using the physical and cyber event relationships has the effect of expanding the knowledge set to gain affirmation or manifestation of suspicions.

The Concept of Minimal Essential

The concept of minimal essential [Mcc01] was developed under USAF Contract F-30602-97-C-0132 and related to the smallest subset of information that would be required, through various formulas and re-computations, to populate an entire data set.

The concept was developed to support the real time recovery of systems so that operational continuity could be maintained. Fundamental to the continuity s the ability of the system to (in real-time) reconstitute data that may have been compromised as the result of an attack.

MEDS in the database context would be the smallest subset of the database that could be used to reconstitute the entire database. The subset could be a specific section of the database or could be a collection of data items across the database. It is a pro-active process where data on a cyclic basis is gathered and then hidden[6] within the system and then made available in the event of a cyber attack. Once an event happens these trusted copies of information are retrieved and used in a series of pre-determined mathematical and logical computations to "fill out" the remainder of the data-base.

Consider, by way of an example, a ground based radar system that is either air defense or air traffic control. Both systems maintain target tracking, with the only differences being in the way the targets are actually processed[7]. Typically, targets are known or unknown, hostile or friendly and the databases that maintain information on these targets are usually fairly large. Should a cyber event wipe out portions or all of the database, we need to reconstitute the database in order to continue with the mission. In this case, hostile targets were considered to be the minimal essential set. The reason for this is straightforward in the sense that a radar system gathers data on every sweep of its antenna. At that time, all targets 'seen' on that scan are correlated to the hostile targets first and then the friendly or unknown targets. If the hostile targets are the ones considered as the MEDS, then the database can be replenished with its hostile targets and then anything 'seen by the radar' that doesn't correlate can be assumed to be friendly or unknown. In any event, the targets of specific interest and the main function of the system can continue without rebooting or restarting.

MEDS doesn't care as much about the value of a piece of data, rather it cares about the relationships that a piece of data has. Those relationships if taken in the aggregate then define the transformation computations for all data in the system. MEDS, as a concept, has great similarities to the mathematical concept of basis vectors. Basis vectors are by definition the smallest set of linearly independent vectors that, taken in some combination, completely span the vector space. It is a similar relationship that MEDS exploits to span the data space of a given system.

The Half Life Concept

The data half-life concept [Mcc01] also emerged under Contract F-30602-97-C-0132 and relates to the time currency of the value of a piece of information. Again using the example of a system given above, consider the relationship between time and target position and target velocity. Systems use time, position and velocity to extrapolate ahead

[6] possibly using steganographic approaches

[7] An air traffic control system is the airborne eyes of civilian aviation. Its function is to maintain separation between civilian aircraft to avoid collisions. Air defense systems seek to bring targets together, so that friendly forces can eliminate hostile forces.

to the next expected target position in order to maintain tracking filters on each target. Using a simple formula of:

$$\text{Target Position}_{t+1} = \text{Target Position}_t + \text{Target Velocity}_t * \Delta t \text{ , where } \Delta t = t_2 - t_1$$

This formula works extremely well if the time intervals are close together, but the value of target position at time t_{12} might not be useful if the system is at t_{144}. This is the essence of the concept of half-life. Similar to the concept of radioactive half-life, information also has a period or window of validity. When combined with the MEDS concept, half-life helps define how current a piece of MEDS really is. In fact, half-life helps define the intervals by which MEDS are gathered and stored.

NEWS 2.0 Overview

The Network Early Warning System (NEWS) is a concept that identifies indicators (conditions and events) that are likely to precede a significant information warfare (IW) attack on network resources. The forecasting system gains network monitoring analysts valuable time to implement countermeasures.

There are some key elements of the NEWS solution and they include: Profiling-Correlation-Clustering that reduces intrusion detection system clutter by using incident profiling features and then associates all the related attack data together Next comes fusion to help resolve the gaps in heterogeneous attack reporting data. Finally there is the forecasting and visualization to cue the feedback both operationally and visually.

NEWS as a methodology was developed under contract for the USAF Information Warfare Battlelab and has been successfully demonstrated. It identifies preparatory activities (called precursors) found primarily at the packet level, that precedes a large-scale attack before any damage is done. NEWS separates large-scale attacks from innocuous activity and then correlates these precursors to stages in a typical attack sequence. It then uses Bayesian networks to estimate the stage of attack that the network is in, and its expected progression, thus characterizing the attacker. As the coordinated attacker moves across multiple sites, the system collects the precursors at a global level, correlates them with the reported intrusions, and produces an estimate of target networks and hosts at risk. Finally, upon identifying precursors to attacks at multiple sites, NEWS applies neural networks to determine whether these precursors are likely to have originated from the same source.

Insider Anomaly Measurement Processing System (IAMPS) Overview

As a result of the rapid growth in computer technology, the government and private sector, has become extremely dependent on automated information systems. Access to these systems may be wanted by individuals or organizations seeking monetary gain, political blackmail or those dedicated to causing damage. As a consequence, access to systems is restricted and controlled to those who are trusted and approved. But what happens when those trusted and approved users begin to use the system for illicit, illegal, or seditious purposes? The number one threat for information warfare attacks for any system, military or otherwise, is the insider attack. The premise has been that adequate perimeter defenses can keep unauthorized users from entering the system through the IP

connection and that means no possible intrusion. Current intrusion detection schemes rely on software checking software for abnormal behavior. This technology has had limited success in the IP based systems environment, but offers no additional protection for non-IP based command and control systems. All this falls short considering the insider attacks. Insider attacks transcend and redefine access control and intruder detection. Addressing the issue of the insider attack, has been more of a post-mortem analysis conducted after the damage has been done. Once a hacker gains access to a system (particularly root access), they have now become a legitimate user and are able to bypass all the conventional intrusion detection concepts. To some degree, this implies that all attacks in some form are really insider attacks. The issue centers on finding ways to decide, in real time, if there is an insider attack in progress.

The IAMPS researched a methodology into detecting, in real time, insider misuse. This program considered the insider misuse to be from user(s) who are masquerading and are not who they claim to be, even though they have passed some level of authentication in the system. Furthermore by looking at several pieces of data (evidence), that inconsistency can be developed. The belief was that users who attempted to gain access under these false pretenses are intentionally malicious.

Succinctly stated, the goal of this research was to gain non-repudiated authentication, and to be able to prove that:

- A person is who they claim to be and we can support that claim empirically through electronic evidence, or,
- A person is not who they claim to be and we can discover the discrepancies using electronic evidence.

IAMPS was a detailed investigation concept of an insider attack detection scheme using fusion that integrates available corroborating information. This approach had three key features:

- First, the overall premise of this approach is that there are measurable system performance parameters. While any one of the individual parameters could fluctuate, several of these being out of "steady state" ranges could indicate unauthorized access or use of the computer system. These steady state measures of system use offer a new and creative method of profiling an attacker based on those measurements. This implies that an additional and empirical measure can be taken and used as "cyber evidence".
- Second, there are other available measurements that exist and when viewed in their entirety, can potentially declare in real time an insider attack. The steady state analysis results can be fused in real-time with timecard, premises, and authentication information, also available in real time, to augment the declaration process and offer fidelity and certainty into the accuracy of the prediction
- Third, this technique detects the computer misuse in real-time without increasing the system overhead or workload.

"Circumstantial evidence is a very tricky thing. It may seem to point very straight to one thing, but if you shift your own point of view a little, you may find it pointing in an

equally uncompromising manner to something entirely different.5[8] " Such are the issues raised as a result of the insider problem. First is to decide what is evidence and what is not. In fact, the approach taken by IAMPS is to consider any piece of information related to authentication as potential evidence. Second is to look at how to gather the evidence. IAMPS addresses this issue by using COTS sensors that already make available electronically the authentication-credential results. Third is a consideration of the privacy issues raised by using certain sensors that may develop profiles of users. IAMPS considers this in the application of the sensors to avoid the negative connotations of profiling. Finally there are the issues of solution impact on overall security in terms of people, process and technology. The IAMPS research was especially sensitive to this point. The IAMPS solution maintains a good balance between the people affected and practicing it, the security processes those people develop and adhere to, and the technologies selected and implemented to enforce and facilitate the security. Achieving that balance is difficult because changes in technology ripple through and affect previously good processes which in turn can affect the operability of the people.

Consider an implementation and some events:

- Unusual amount of outgoing email at 1a.m. from director
- Email alert "triggers" defense system
- Email analysis system detects classified documents in the stream
- Network log-on shows managing director logged into system
- Premises entry devices show director not on site
- Timecard accounting system shows director on holiday

The network monitoring shows an unusual amount of outgoing email at 1am from a director. This in turn generates an Email alert that “triggers” the defensive system. The Email analysis system detects classified documents in the email stream. Electronically accessing the log, it is found that the director is in fact logged on to the system. However, the Premises entry devices show that the director is not even on site. By way of a cursory check, the timecard accounting system is checked and it shows director on holiday. In essence, we have achieved the 1st Level of detection - Who it Isn’t. This allows us to take immediate action such as capturing the forensic evidence, terminating the connection and dispatching security to the location.

Summary

In conclusion, fusion represents if not the strongest, then one of the strongest approaches to correlation of events. That includes BOTH insider and outsider attacks. And even more importantly, there is the importance of doing this detection in real time. The ability to use information already existing to help protect a system is powerful. Real time allows design for courses of action to become proactive instead of reactive. Picking up the pieces after the damage is complete may be impossible and have extreme implications, such as the case of Omega engineering.

---

[8] From the website of Sherlock Holmes quotes, www.bakerstreet221b.de/canon/index.html. and the novel The Boscombe Valley Mystery

These approaches show a good balance between people, process and technology. Maintaining that balance while expanding protection will continue to be a challenge. Fusion as applied in this domain is relatively recent there are some key issues that remain. First is the use or invention of additional sensors that may be more intrusive or perform different types of profiling, keystroke monitoring being a notable example. Once the detection begins, there is the problem and legal and custody issues surrounding evidence gathering and preservation. And finally, there is the fact that the insider timeline is substantially short and requires automated assistance.

References

1. Casey, E. (2001), Handbook of Computer Crime Investigation, Academic Press, Bath, Great Britain
2. Fox-Davies A.C. and Carlyon-Britton P.W.P. " A treatise on the law concerning names and changes of name", Elliot Stock, London 1906.
3. Kruse, W., Heiser J.(2001), Computer Forensics Incident Response Essentials, Addison-Wesley, Boston, Mass.
4. Lindqvist, U. and Porras, P., "Detecting Computer and Network Misuse Through the Production Based Expert System Toolset (P-BEST), Proceedings of the 1999 IEEE Symposium on Security and Privacy, Oakland, C., May 9 - 12, 1999
5. McCallam D., Piggott C., Newland R., Rapid Recovery of Information for Real Time Intruded Systems, AFRL-IF-RS-TR-2001-55, Rome, New York
6. McCallam D., Newland R., Jajodia, S., Wheeler, A. Demonstrating Information Resiliency, AFRL-IF-RS-TR-2001-156, Rome, New York
7. D'Amico, McCallam, and Zavidniak, "Achieving Information Resiliency", Published by Elsevier Science (UK), Information Security Technical Report on Information Warfare, Vol. 4, No 3; London, United Kingdom (August, 1999)
8. Nuemann, P., "The Challenges of Insider Misuse", Prepared for Workshop on Research and Development Initiatives Focused on Preventing, detecting and Responding to Insider Misuse of Critical Defense Information Systems, Santa Monica, Ca., 16-18 August 1999
9. "Research and Development Initiatives Focused on Preventing, detecting and Responding to Insider Misuse of Critical Defense Information Systems Results of a Three Day Workshop, Santa Monica, Ca., 16-18 August 1999
10. Song, D., Wagner, D., and Tian, X., "Timing Analysis of Keystrokes and Timing Atacks on SSH", Presentation to the 10th Usenix Security Symposium, Washington D.C., August 2001
11. Wood, B. "An Insider Threat Model for Adversary Simulation", Appendix B from the Workshop on Research and Development Initiatives Focused on Preventing, detecting and Responding to Insider Misuse of Critical Defense Information Systems, Santa Monica, Ca., 16-18 August 1999

This page has been deliberately left blank

Page intentionnellement blanche

Click here to view PowerPoint presentation; Press Esc to exit

Secure Shell Proxy Intrusion Detection

Marc Plaggemeier, Jens Tölle
University of Bonn
Roemerstrasse 164, D-53117 Bonn, Germany

{plaggeme,toelle}@cs.uni-bonn.de

Abstract:

This paper describes the prototype system of a secure shell proxy intrusion detection system (called EPIDS) that controls SSH-connections between clients and servers. EPIDS is a transparent proxy which makes use of a man in the middle attack. Every connection between client and server is decrypted, inspected and encrypted again. If the system finds some unusual behavior it is able to close the connection.
The system includes misuse and anomaly detection components. It learns the "normal" user behavior by studying the commands executed by every user and stores them in a data structure called "sequence tree". We describe a new approach to detect anomalous behavior by comparing user commands with normal user behavior stored in such a tree. We also describe which kind of attacks the system is able to detect and its limitations.

I. Introduction

Network based Intrusion Detection approaches have a major drawback. It is not possible to supervise the contents of encrypted communication channels. These channels can only be supervised by host based ID systems, making ID components necessary on all hosts.

Our approach is a solution for secure shell connections (based on the source code of openssh version 2.1.1p1 [SSH2002], [SEC2002]) using password authentication. Usage of RSA or DSA authentication is not supported in the prototype system. The basic idea is to use a transparent proxy. The proxy accepts connection requests from SSH clients and forwards data to the selected SSH-server.
The system is called EPIDS (real-time proxy intrusion detection system).

After establishing the SSH connection, the system uses a database system to check

- each command line,
- directory outputs,
- text file outputs, and
- compiler outputs.

All these checks are done in real-time. The method allows the detection of command line oriented attacks. Known attack tools, mainly used by script kiddies, can be discovered.

The real-time checking allows the proxy to shut down the SSH connection after the recognition of the beginning of an attack and defend the server from an ongoing attack.

Paper presented at the RTO IST Symposium on "Real Time Intrusion Detection", held in Estoril, Portugal from 27-28 May 2002, and published in RTO-MP-101.

The systems consists of both misuse and anomaly detection components. The prototype implementation of the system detects misuse pattern stored in a database:

- access to "forbidden" files and directories
- usage of "forbidden" commands
- compilation of "forbidden" files

The system can even discover critical contents of files being edited. Besides these misuse detection approaches, some anomaly detection components are integrated: The system learns user specific behavior. Which commands does a user execute, and especially, user specific typical sequences of commands? Users have characteristic behavior in command usage. The method of sequence trees stores information on probabilities of one command following another. The root of the tree structure is a command given by a user, the sons of the node represent succeeding commands, edges are weighted with the corresponding probability. New command sequences are looked for in the given sequence trees. The depth reached in the tree by following the path from the root (representing the first command) to the succeeding nodes is a value indicating the probability of the sequence for the specific user. Unusual command sequences are easily recognized as an anomaly.
This anomaly detection idea is not limited to network based intrusion detection systems. It can easily be used in host based systems as well. The usage of additional ID components in the proxy is possible as well.

II. EPIDS Overview

The basic idea of EPIDS can be described as: *Look at the screen of the user*. The system controls everything the user sees on his screen and tries to find out if it is an attack or not. The general idea of this approach is not new, but up to now this approach was used with unencrypted connections only. The only way to inspect a secure shell connection is a proxy which decrypts the connection, inspects the unencrypted data and encrypts it again to forward the packet to the SSH server.

How does EPIDS work?

As described above, EPIDS is a transparent proxy. The user does not notice that his SSH connection is controlled by EPIDS. This is only possible if the utilized authentication method is password driven and not DSA/RSA. When using DSA, SSH uses a local copy of the public key to check if the authentication is correct. The client sends the public key to the server. This public key is signed with the client's private key. The packet sent to the server also contains a unique identifier of the SSH connection. Thus, it is not possible to forward the public key of the client to the server. The only way to use DSA authentication is to send the public key of the proxy to authenticate the client to the server.
When the connection is established, the user uses the SSH connection by executing commands and the server sends his output to the client. Every data packet flowing between client and server is inspected.
EPIDS consists of two parts. The first part is the transparent proxy which controls the data flow. The second part is the "profile daemon" which controls the commands (anomaly detection) and tries to find out if there is abnormal behavior.

- The transparent proxy:

  The major tasks of the transparent proxy are connection management (establishing and closing a connection) and the inspection of the data flow. Figure 1 shows the behavior of the proxy. One part of the transparent proxy is the "misuse thread". This thread inspects the output of the server. It is looking for known source codes, directories and compile operations. If it finds suspicious behavior, it can initiate the shutdown of the connection. Details of the misuse detection are described in the next section. An other part is the anomaly-thread which checks if the profile daemon has found something suspicious. Both the misuse and the anomaly thread can initiate the shutdown of the connection.

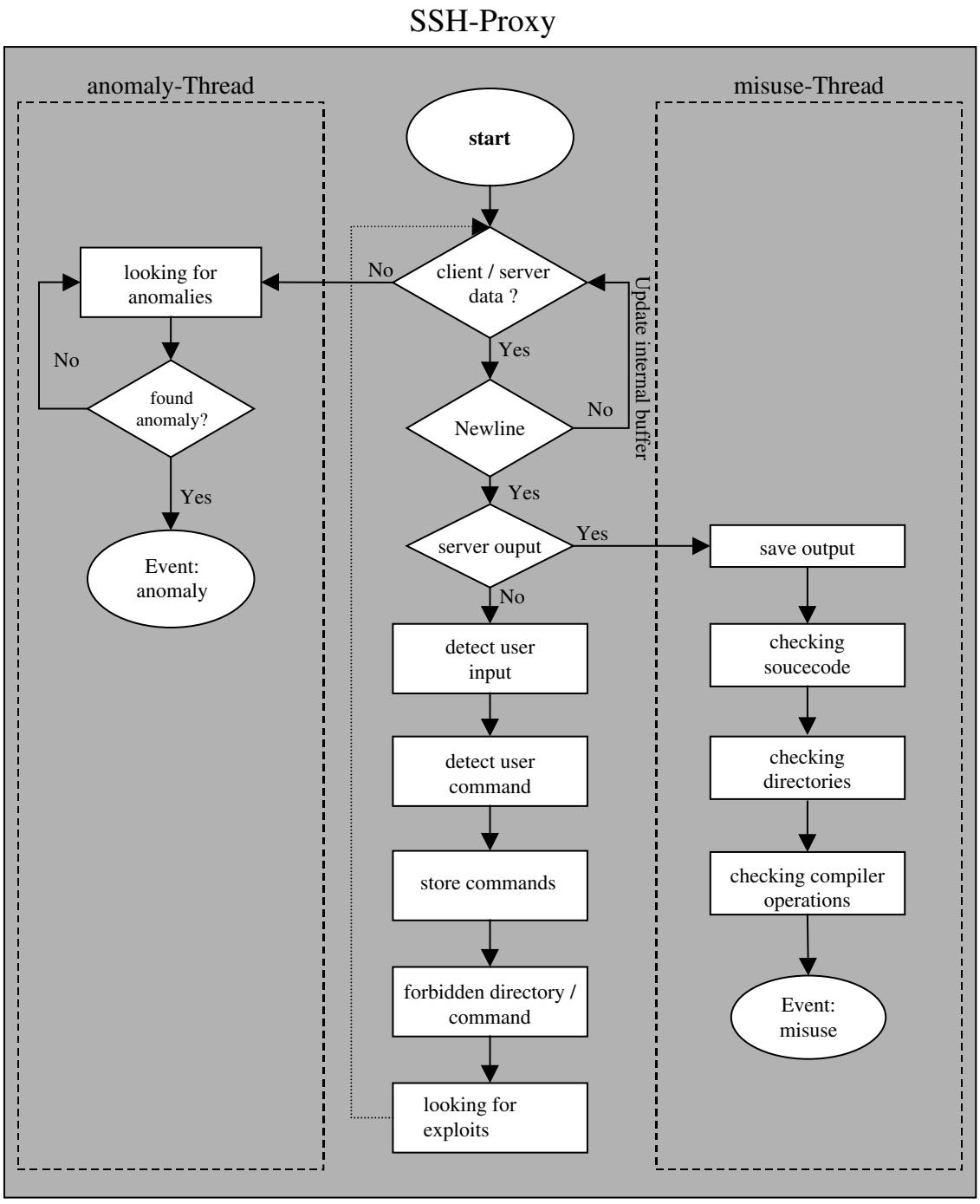


Figure 1: SSH proxy overview

- The profile daemon:

  The profile daemon is responsible for the anomaly detection. It has access to the database with the collected information on the normal behavior of the user and compares the normal behavior with the current behavior of the user. In the case of the detection of abnormal behavior, it can initiate the shutdown of the connection as well.

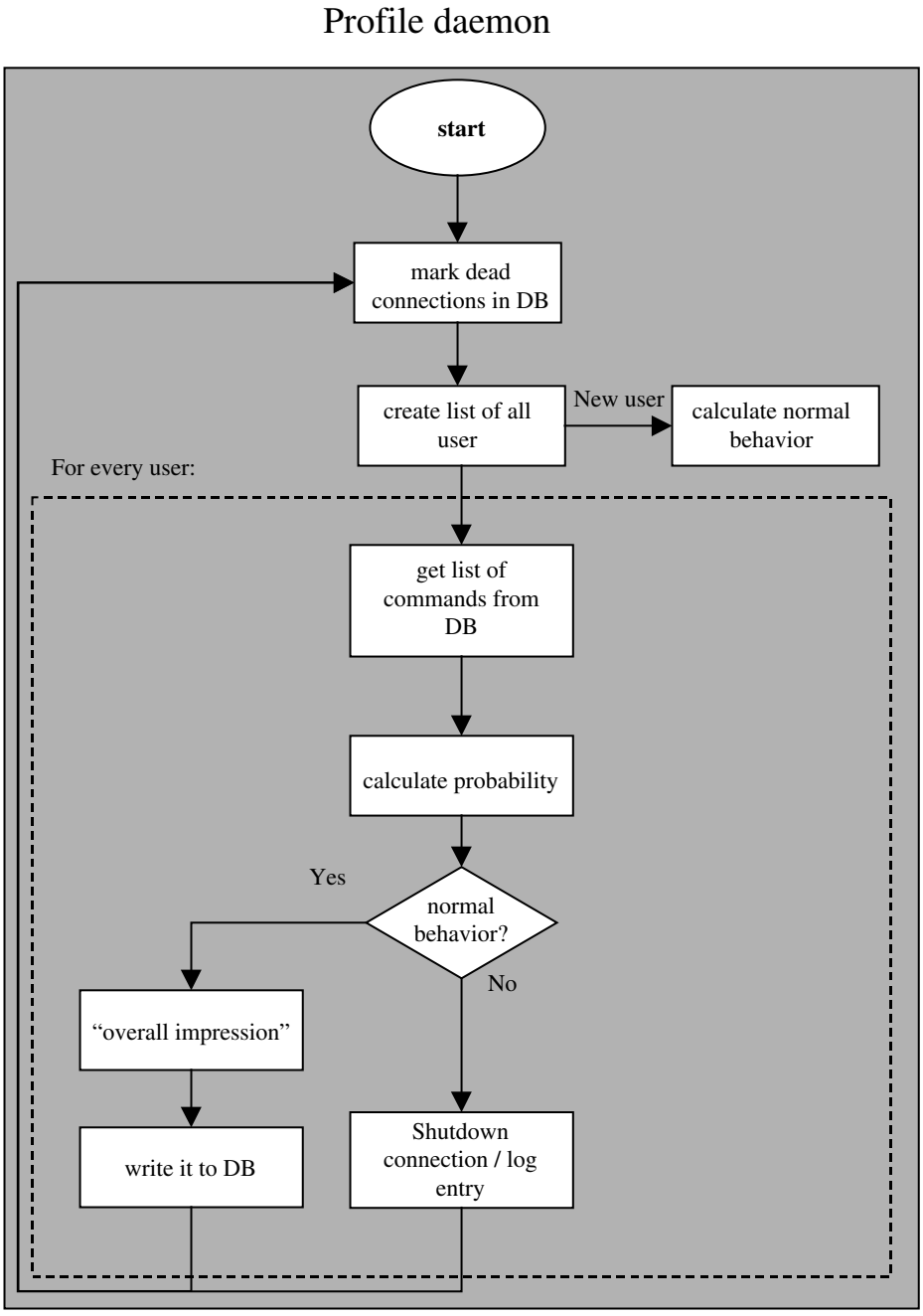


Figure 2: profile daemon behavior

III. EPIDS intrusion detection

This section describes the different anomaly and misuse detection approaches integrated in EPIDS. First we present three different methods of anomaly detection followed by a description of the misuse detection component implemented in EPIDS.

III.1 EPIDS anomaly detection

We have implemented three different approaches to anomaly detection. The first one is based on the work of Dorothy Denning (IDES [DED1987]). It uses "time slots" to detect anomalous behavior. This approach will not be described in this paper. The second one uses homogenous markov chains to detect anomalous behavior. The last method is a new approach which is based on empirical data.

Anomaly detection using "homogeneous markov chains":

A method to detect anomalous behavior is the use of "homogenous markov chains". Nong Ye developed an IDS which is based on markov chains (see [NY2000]). The advantage of such an approach is the independence of time. To every pair (E_j, E_k), we have a corresponding condition probability p_{jk}. Given that event E_j has occurred, the probability of E_k at the next command is p_{jk}. A simple table can be used to realize homogenous markov chains. The next table illustrates an example.

	Ls	cd	...	rm
ls	0.2	0.1	...	0.05
cd	0.3	0.15	...	0.01
⋮	⋮	⋮	⋮	⋮
rm	0.03	0.1		0.3

EPIDS uses different methods to detect anomalies.

Probability of sequences:

With the aid of the *probability of sequences* we calculate the probability that a user executes a sequence of commands. The probability is defined by:

$$P(X_0 = i_0, ..., X_n = i_n) = \pi_{i_0} p_{i_0 i_1} p_{i_1 i_2} ... p_{i_{n-1} i_n}$$

$\pi_i = P(X_0 = i)$ is the probability of the initial command.

There are some disadvantages. For example:

- The probability is getting lower when calculating large sequences.
- If the sequence includes an unknown subsequence the probability of the sequence is 0.

The latter one can be prevented by defining a minimum probability for an unknown command.

Mean value:

A simple method to detect possible anomalies is the *mean value* of a sequence. The mean value is calculated by summing up the single probabilities and divide the sum by the number of executed commands. A disadvantage of the mean value is that low probabilities may be covered by high probabilities. Consider the sequence “0 - 0 - 0.2 - 1 - 1”. The *mean value* of this sequence is 0.44. This could be regarded as a “normal” value, but 3 of the 5 commands were improbable or unknown.

Threshold values:

The probability of every single command will be compared with a threshold value. The occurrence of suspicious commands will be reported.

Anomaly detection using “sequence trees”:

A new idea invented to detect anomalous behavior is the use of sequence trees. The basic idea is to identify sequences of commands which are typical for a user. The suitability of this idea is shown in (TIM [TSL1990]). New is the method EPIDS uses to store the command sequences and the usage of this data structure for the detection of anomalous behavior. Let us first define a sequence:

A **sequence** is a series of successive commands executed by a user during an SSH-Session. The sequence is beginning with the input of a command and ends if the user repeats this command or if the length of the series has reached a given threshold.

These sequences can be stored in sequence trees. To create sequence trees, the system scans all executed commands. For each of the commands used during an SSH session, a sequence tree is created. The following figure illustrates an example of a sequence tree. The root of the tree structure is a command given by a user, the sons of each node represent subsequently used commands.

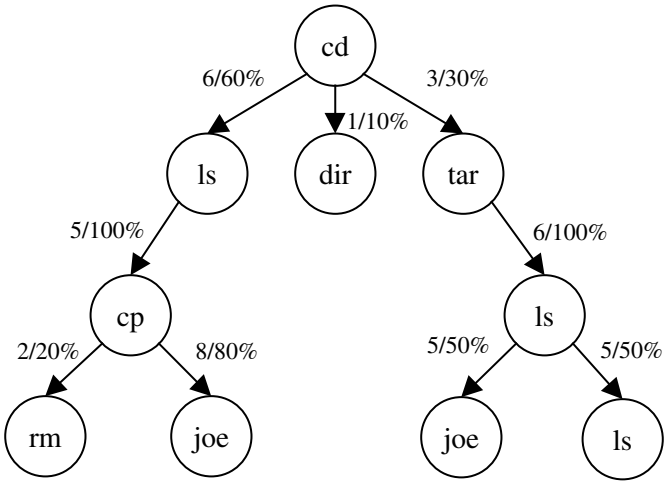


Figure 3: Sequence tree – example

How can we detect anomalous behavior?

This approach is not based on any probability measure. Instead of this, the system is looking for known command sequences in sequence trees. The average depth reached in all sequence tree searches is a measure to detect anomalous behavior.

We give an example to illustrate this approach. See the figure above. The sequence "`cd - ls - cp - joe`" reaches the depth 3. Sequence "`cd - cp - joe`" reaches depth 0 (the depth of the root is defined as 0). The system calculates the average depth of all sequence trees, not only the average depth of the current used sequence tree. Thus, at the beginning of a SSH session there might be a high deviation. But after a few commands executed by the user, the deviation decreases.
A low average acts as an indicator for anomalous behavior. The behavior is treated as anomalous behavior if the average depth is lower than a given threshold value.

Another method to detect anomalous behavior is the use of probabilities given in the sequence tree like TIM does. We can simple multiply the probabilities of a sequence and test if it is lower than our threshold value. Improbable sequences can be removed from the sequence tree. Only sequences with high probabilities remain. Rules like cmd1 – cmd2 -> (cmd3: 80%, cmd4: 20%) can be added and tested.

III.2 EPIDS misuse detection

Several approaches of misuse detection were developed and tested. Only approaches which use name and parameters of the executed commands and the server output could be used. So attacks of the following kinds can be detected:

- Exploits executed via the command line.
- Inspecting the server output: Source codes, directories and compiler outputs.
- Usage of forbidden commands, access to forbidden directories and files.

Detecting exploits:

Some attacks can be described by a list of commands which are executed to gain root access to a system. The execution of these commands must be ordered to be successful. So a successful attack can be described in form of a table. A well known attack is a security hole in old versions of sendmail. The attack looks like that:

```
1. cp /bin/sh /var/mail/root
2. chmod 4755 /var/mail/root
3. touch x
4. mail root < x
5. /var/mail/root
```

Line 3. is not important to gain root access. As well the filename is not important.
So our table looks like this:

Line	Directory	Command	Parameter	Predecessor	Weight
1	/var/mail/	cp	/bin/sh	0	20
2	/var/mail/	chmod	4755	1	30
3		mail	root	2	20
4		/var/mail/root		3	20

The system compares each command with the stored attacks before forwarding it to the server. When it finds a matching entry it writes an entry to an internal database table. Attacks which were identified are stored in this table. Thus, the system can detect distributed attacks. Attacks which are executed over a large period of time can also be detected.

After writing the entry to the internal table it sums up all entries to compare the total weight with a threshold value. The weights found in the table and the threshold are administrater defined values. If the overall weight exceeds the threshold value, the connection can be closed by the proxy.

Inspecting directories:

Every time the user executes `ls`, `dir` or some other command to see the content of a directory the output of the server is inspected. The filename is extracted from the output and compared with the database. Each filename can be associated with a directory name. Whenever the system detects an directory – like a rootkit or something similar – it is possible to close the connection.

Inspecting files:

When the user executes a text editor like `joe` or `vi` EPIDS checks the server output to detect well known attack tools – like shell scripts. Every line is compared with the database. If a threshold value is exceeded it is possible to close the connection.

What kind of intrusion does EPIDS detect?

EPIDS is able to detect intrusions which are accomplished by "shell commands". Attacks which uses scripts can be detected by inspecting the files. But the attacker does not need to open the file and he can modify the source code of the attack to mask it. A script like

```
#!/usr/bin/perl
$attack = "ec)ho th-i-s i+s a a%tt(ack; r!m /t-mp/att+a-!ck";
$attack =~ s/(\(|\)|\+|\-|\%|\!)//g;
system ("$attack");
```

cannot be detected.
Another problem is the order of the commands used by a attack. Most of the attacks can be executed in similar ways. The name of the commands are not unique. Look a the following example.

```
#!/usr/bin/perl

cd /tmp
ln –s /bin/rm remove
remove /tmp/attack
```

As well as any other misuse detection system it is only possible to detect intrusions which are stored in the database. New intrusions must be entered in the database by the administrator.

IV. Evaluation

In this section we describe practical results. First, we introduce the utilized hardware. Thereafter, we present the performance tests. The most important part of this section is the evaluation of our different approaches to anomaly and misuse detection.

Used Hardware:

EPIDS was implemented on a Linux System using Kernel 2.4.0. It runs on an Pentium III Processor (500 MHz) with 128 MB RAM. The used database was a MySql [MYS2002] Database version 3.23.38 which is installed on the same host.

IV.1 Performance test:

We tested 20 concurrent connections. After establishing a connection the program `top` was started to produce a constant data flow. Furthermore the directory `/usr/bin` was dumped regularly. Directories are also inspected by EPIDS. Every packet tested by EPIDS stressed the server.
Every connection used 1.4 MB RAM. The load average of the CPU was 20 to 30 percent. The maximum load was 40 percent. Therefore, the system was not overloaded.

IV.2 Profile characteristics:

Because of the protection of privacy only volunteer users collected data.

Here is an overview of the collected data.

User	Time interval	# commands	# Distinct commands	# sessions
A	1 day	285	44	3
B	38 days	3096	142	353
C	1 day	252	24	4
D	40 days	4132	110	617
E	29 days	1217	32	18
F	21 days	407	38	4
G	14 days	114	25	19

IV.3 Anomaly detection:

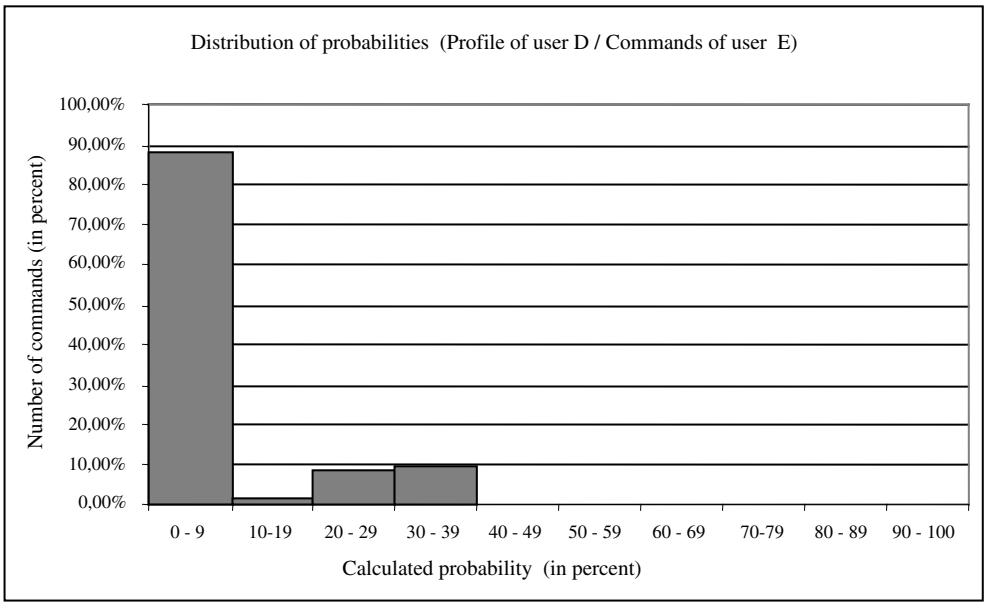
Two different kind of tests were performed to test the anomaly detection.

- The first test checks if the system detects anomalous behavior. The system first learned the profile of a user. Then commands of a different user were executed.
- The second test checks if the system detects normal behavior. The goal of this test was to verify if the system produces a false alarm.

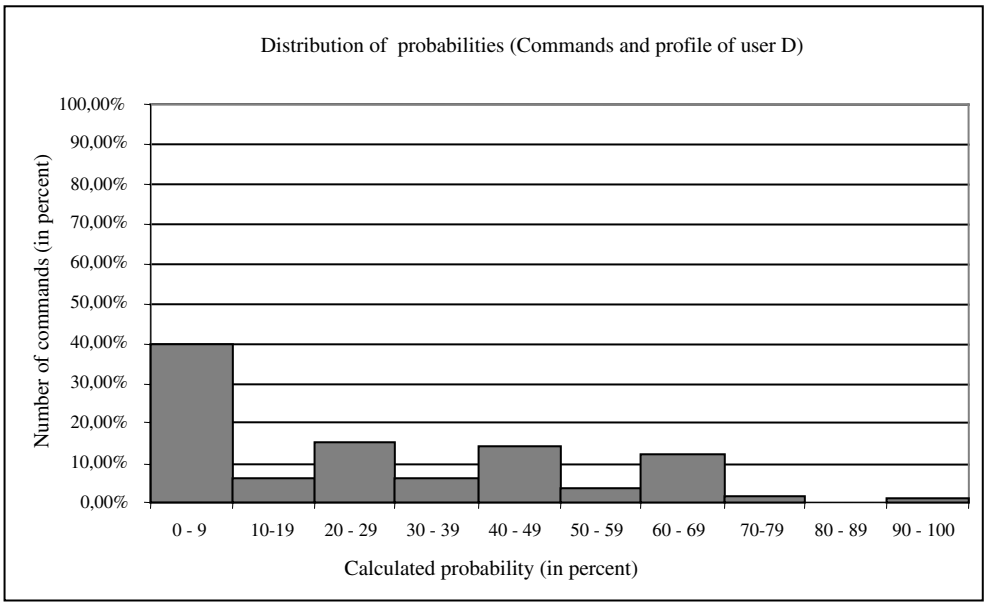
Anomaly detection using “homogeneous markov chains”:

The following tests are to give information about the usability of the markov chain model. Above we described three different methods to detect anomalous behavior. Each of these methods were tested.

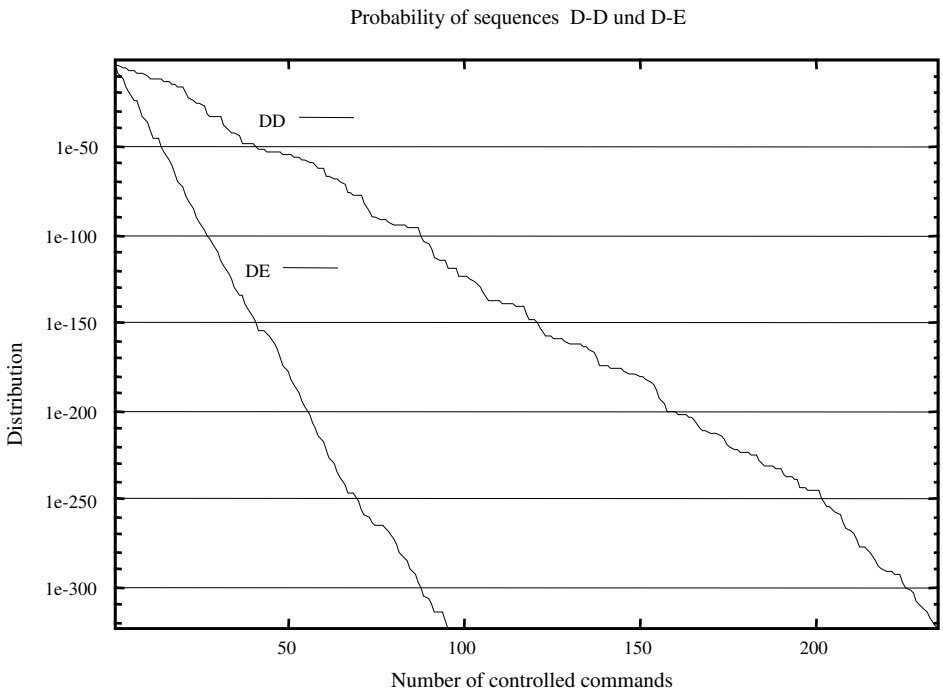
First the system learned the profile of user D. After that, commands of user E were executed to test the system. The following figure shows the distribution of the calculated probabilities.



Second test: The system learned the profile of user D. To test if the system is able to identify normal behavior we executed command sequences of user D. Here is the result:



An other method to detect anomalous behavior is the probability of sequences. The next figure illustrates that there is a difference between anomalous and normal behavior.



Anomaly detection using “sequence trees”:

This section describes the new developed approach to detect anomalous behavior. The first table shows the result of our first test. The system learned the profile of user D, commands of user E were executed.

Then we executed commands of user D to test if the system detects anomalous behavior (results in the second table).

Here are the results:

Description	Absolute value	Value in percent
Number of commands checked	1073	100 %
Average depth	**0.0233**	
Number of cmds – depth 0	1051	97.95 %
Number of cmds – depth 1	19	1.77 %
Number of cmds – depth 2	3	0.28 %
Number of cmds – depth 3	0	0.0 %
Number of cmds – depth 4	0	0.0 %
Number of cmds – depth 5	0	0.0 %
Number of cmds – depth 6	0	0.0 %
Number of trees not found	**640**	59.65 %
Number of commands checked	950	100 %
Average depth	**1.42**	
Number of cmds – depth 0	146	15.37 %
Number of cmds – depth 1	486	49.26 %
Number of cmds – depth 2	200	21.05 %
Number of cmds – depth 3	86	9.05 %
Number of cmds – depth 4	31	3.26 %
Number of cmds – depth 5	15	1.58 %
Number of cmds – depth 6	04	0.42 %
Number of trees not found	**14**	1.47 %

The next table shows results from the comparison of the other users. Perhaps the similarities between user D and user F are because of a sequence which both user executes. The sequence "`cd ls`" is often executed by both users.
The fraction of this sequence in the data of user F is high. If we compare the profile of user F with the commands of user D the average depth is only 0.24921.

Profile of user	Commands of user	# commands	Average depth	# trees not found
B	A	246	0.23984	110
B	B	418	0.66029	84
B	C	216	0.04167	169
B	D	1152	0.00955	876
B	E	1192	0.00252	1112
B	F	381	0.11024	187
B	G	104	0	78
D	A	244	0.37705	88
D	B	2172	0.22744	932
D	C	196	0.09694	79
D	E	1073	0.02330	640
D	F	362	0.64365	77
D	G	98	0.32653	40
F	D	3784	0.24921	2122

IV.3 Misuse detection:

Identifying exploits was tested by executing programs with well known bugs like `sendmail` or `mailx` bugs. All exploits were identified by the system.

Inspecting files:

There is one major drawback when inspecting files. Many files have lines like "`#include <stdio.h>`". Those lines are not significant to identify an attack. So we have to optimize such files and remove lines which are not important to implement the attack. The next table shows our results testing the file `statdx.c`. Without optimization many other files were identified party by mistake. With optimization the results were much better. Results can be seen in the following table, which contains a list of attack codes. The misuse detection component was looking for the source of statdx.c.

Sourcecode	Without optimization	With optimization
statdx.c	**74.70 %**	**100.0%**
Xxploit.c	7.25 %	1.96 %
Bashack.c	5.36 %	2.13 %
Cwdtoolsexp.c	15.38 %	0.0 %
fdmnt-smash2.c	7.25 %	5.56 %
Fdmount_xplt.c	20.00 %	0.0 %
gnomelib.c	10.87 %	3.45 %
restore-exploit.c	5.30 %	3.42 %
rpc-statd-xpl.c	8.59 %	2.54 %
statd-toy.c	11.39 %	3.07 %
susekill.c	22.22 %	2.78 %
wuftpd-2.6.0-exp2.c	5.09 %	2.55 %
tsig.c	14.23 %	9.09 %
fd-ex.c	17.07 %	3.13 %
sysctl_exp.c	11.54 %	4.17 %

V. Conclusion

In this paper we introduced a secure shell proxy intrusion detection system called EPIDS. EPIDS checks decrypted SSH connections between clients and servers and tries to find known attacks as well as anomalous behavior. We introduced a new approach to anomaly detection. This approach uses sequence trees to compare normal behavior with the current behavior of a user. The performed tests have shown that this approach is suitable to detect anomalous behavior.
We also mentioned the misuse detection components integrated successfully and explained limitations and their reasons.

The two described approaches to detect anomalous behavior - markov chains and our new approach using “sequence trees” - are both suitable.

Markov chains:

There is a remarkable difference between anomalous behavior and normal behavior. But using threshold values to detect anomalous behavior is not applicable because of the distribution. It is noticeable that the probability of nearly 40 percent of the executed commands is lower or equal than 10 percent. Thus, using threshold values may cause many false alarms.

Sequence trees:

Our new developed approach using sequence trees is able to detect anomalous behavior. There is a remarkable difference between the average depth of anomalous behavior and normal behavior. In all tests the average depth of an anomalous behavior was lower than 0.7. Seven of twelve values were even lower than 0.1. These results makes us confident that this approach is suitable for detecting anomalies. Additional results with larger user populations are necessary to draw conclusions on the suitability in larger environments.

Our approach is not limited to EPIDS or other network based systems. It is possible to use this approach in host based intrusion detection systems (for example C2-Audit [DOD1985],[DOD2000]) as well. This would have an additional benefit: The inspection of every command used, not only those who where visible on screen.

VI. References

[DED1987] Dorothy E. Denning
An Intrusion-Detection Modell
IEEE Transaction on Software engineering, 13: pages 222-232 February 1987

[DOD2000] Department of Defense
Rainbow Series
http://www.radium.ncsc.mil/tpep/library/rainbow

[DOD1985] Department of Defense
Trusted computer system evaluation criteria, December 1985
http://www.radium.ncsc.mil/tpep/library/rainbow/5200.28-STD.html

[MYS2002] MySQL Database
http://www.mysql.com/

[NY2000] Nong Ye.
A Markov Chain model of temporal behaviour for anomaly detection.
Proceedings of the 2000 IEEE Workshop on Information Assurance and Security,
6. July 2000. United States Military Academy, West Point, NY.

[RGB2000] Rebecca Gurley Bace
Intrusion Detection
Technology Series Macmillan Technical Publishing 2000

[SEC2002] Homepage of the IETF Working Group Secure Shell
http://www.ietf.org/html.charters/secsh-charter.html

[SSH2002] Authors of SSH. Manualpages and Drafts of SSH
http://www.openssh.com/

[TSL1990] HS Teng, K.Chen, SC Lu
Adaptive real-time anomaly detection using inductively generated sequential patterns.
Technical report, DEC, 1990

Click here to view PowerPoint presentation; Press Esc to exit

Securing Mission-Critical Core Systems

George Valvis
Expertnet S.A.
244 Kifisias & Achilleos 1
Athens, Greece, GR- 15231

Tel: 003-01-6785000
e-mail:gval@expnet.gr

Panagiotis Sklavos
Department of Electrical and Computer Engineering
National Technical University of Athens
9 Heroon Polytechneioy
Athens, Greece, GR-15780

Tel: 003-01-7721536
e-mail:psklavos@softlab.ntua.gr

Despina Polemi
Institute of Communications and Computer Systems
National Technical University of Athens
9 Heroon Polytechneioy
Athens, Greece, GR-15780

Tel: 003-01-7722466
e-mail:polemi@softlab.ntua.gr

Abstract: Despite the aggressive growth of internet technologies, the vast majority of today's sensitive information is still held, maintained and processed in the context of mainframes and legacy systems. However, before e-business can utilize the intelligence inherent in the huge corporate databases, the challenge of the secure integration with existing core systems must be pursued. In this paper we present an architecture for integrating securely legacy databases into state-of-art e-business applications preventing both internal and external misuse. The paper sets the technical criteria on which a sound solution must be based on, with respect to security, scalability and availability. The paper outlines a core foundation, which delivers many security-enhancing technologies and techniques within the network, host, database server, and application layer. These technologies and techniques will provide the necessary functionality for embedding **authentication**, **authorisation** and **accounting** (**AAA**) mechanisms into the legacy platform. The scalability and availability requirements of the approach are also presented. The technical solution focuses around the content provider model and presents how access to sensitive corporate information can be controlled, used and monitored in the context of AAA mechanisms. Special emphasis is provided to the design of a system that recognizes near real-time intrusion attempts based on a defined normal behaviour model.

1. Introduction

Despite the aggressive growth of internet technologies, the vast majority of today's corporate information is still held, maintained and processed in the context of mainframes and legacy systems. The business intelligence stored in the core corporate systems has been built up gradually over the years and now constitutes the basic fuel that drives the operation of large companies. The core systems have evolved into mission-critical systems supporting the whole range of business operations and serving as a valuable tool for business development. On the other hand, e-business has become the new default standard for the new era of commerce minimizing return-of-investment time periods. E-business systems are usually set up within very tight timescales requiring very different design philosophies and human skills than their legacy counterparts. Being so distinct and different environments, the legacy systems and e-business systems have become a necessity in today's business world and therefore their tight integration is a requirement.
However, before e-business can utilize the intelligence inherent in the large corporate databases, the challenge of the secure integration with existing core systems must be pursued. The common e-business environment operates in an open manner via Internet and this brings forth new security challenges. Moreover, the legacy systems constitute a wealth of information for potential attackers both from the inside and the outside networks[1]. In this paper we attempt to provide a framework that addresses the trusted insider misuse threat. This threat involves normally authorized users who are considered trusted by the system and attempt to escalate privileges by taking advantage of weaknesses of the system in order to gain access to sensitive information they don't have authorization for. The paper describes an overall platform that accommodates an

Paper presented at the RTO IST Symposium on "Real Time Intrusion Detection", held in Estoril, Portugal from 27-28 May 2002, and published in RTO-MP-101.

enhanced Authentication, Authorization and Accounting engine capable of diagnosing and responding to near real-time misuse attempts. Special emphasis has been given to providing such a design that allows seamless integration to the existing mission-critical systems, securing them at the same time, so that normal business operations are not disturbed.
The rest of the paper is organised as follows: in section 2 we set out the criteria on which we base our proposed security framework and give the necessary requirements that must be met by the system. Section 3 gives the overall technical framework of the solution describing the model on which the solution is built, the basic components and sub systems. Special emphasis has been given to the description of the near real-time misuse detection subsystem which fortifies the operation of the platform. The paper ends with conclusions.

2. Criteria

In order to design a platform that will be capable of meeting the business requirements of an organization heavily dependent on legacy mission-critical systems for its business continuity, we first define the major factors that could most severely affect the authorized access to sensitive information. The lack of infrastructure security can lead to unauthorized transactions, which could expose the business to the risk of insider misuse, leak of information, data modification or replacement, false representation and service interference. In addition to those security threats, scalability may also have an impact on the secure, responsive access to this mission-critical service. Furthermore, lack of service availability may also be considered as a damaging and visible barrier to business continuity and customer satisfaction. Finally, it is an absolute requirement that the existing application logic as well the data contained within the legacy database be preserved and maintained. We define the terms of security, scalability, availability and preservation of existing application logic and data as criteria that act as a roadmap for the specification of a new system design that will allow regulated and controlled access to the sensitive information.

2.1. Security

The set up of a secure system infrastructure comprises both security policy issues as well as technology issues that apply countermeasures against the major threats to security. The inability to provide a secure infrastructure for the viewing of sensitive corporate information opens the business up to negative consequences in the form of insider misuse and leak of information. Lack of security can lead to customer dissatisfaction, could harm the business reputation and provoke legal issues. Legacy systems and databases are vulnerable to security breaches because of their complex nature, insecure password mechanism, misconfigured operating systems or unrecognised system backdoors. To reduce the risk of these vulnerabilities an organization should apply the following general security principles when designing regulated and controlled access to the legacy systems:

- Less privilege - a user should only be granted those privileges that she needs in order to perform the required job function;
- Defense in depth - multiple layers of protection[2];
- Accountability – although the ultimate goal is the prevention of security breaches, the ability of the security solution to provide the means of detecting abnormal user behaviour, is critical;
- Encryption should be used whenever possible[3,4]; and
- Clear, defined security policy and procedures.

2.2. Scalability

Scalability refers to the ability to grow the business process without any one component of infrastructure being a limiting factor. Scalability is measured in terms of how quickly and adequately infrastructure can grow while maintaining acceptable end-user performance and availability. As a general rule, building a platform that can quickly and adequately scale throughout the solution without slowing the business is key to ensuring time-to-market and effectively managing growth and change. It shouldn't be overlooked that ensuring seamless, adequate scalability can also affect availability in terms of prolonged planned downtime and poor performance. Scalability of the overall business depends upon having not only the most scalable components in the solution, but also an architecture that allows seamless scalability through planning and best practices.

2.3. Availability

Availability is defined as the uptime of whole application system and network systems as perceived by end-user application availability. In other words, the availability is examined only in terms of continuity of service, whereas the data durability is not considered as a requirement.

3. Overall Technical Solution

In order to provide a solution for the aforementioned security challenge we adopted a methodology which combines the following generic approaches for IT security provisioning:

1. The definition of an appropriate administrative schema that determines the followed procedures to access the sensitive corporate information. The administrative schema is directly derived by the security policy[4].
2. A building-block approach for designing a security solution that utilizes various technologies in multiple stages in order to provide authentication, authorization and accounting mechanisms upon accessing sensitive information within the defined administrative schema.
3. Exercise Defence in Depth as a general principle, in order to apply several layers of defense, sometimes overlapping and achieve the broadest and most complete coverage of the content provider platform[2]. This would be accomplished utilizing diverse methods and technologies under the unified umbrella of a comprehensive security policy.
4. Additionally, an automated reporting procedure has been suggested that will help indicate any behaviour that deviates from the one imposed by the security policy, thus providing near real-time misuse detection support for the overall platform. For defense in depth to work effectively, auditing information could be correlated before being analyzed and aggregated in order to provide a complete platform-wide view of the security posture.

3.1. Security Policy

The next section describes the security policy that we have specified in order to support the operation of the security solution. Usually in legacy systems, all personal information about customers and their profiles are kept in distinct databases, which are stored in a mainframe and are managed by a proprietary database application. This information is characterised as sensitive. A large user community access the legacy system's database and have the necessary privileges to view, download or use any ad hoc reporting tool for acquiring knowledge of sensitive customer data. A new security policy should be stated that would have these privileges permanently revoked by disabling the corresponding attributes in their user and group profiles in the database and the underlying OS of the legacy system. These users will still be able, though, to connect to the database and perform data manipulation operations on non-sensitive information. The security policy involves the set up of a dedicated department within the organization that hosts the legacy system, which will be responsible for providing both customers and commercial partners representatives with the information they need after having removed its sensitive nature. This special department, that in this document will call "Personal-Data Processing Team" (PDPT) will be servicing requests originated from the organization's customer care departments and the organization's external commercial partners. The operators of this department will constitute the only user group within the organization, privileged with the necessary attributes to access the sensitive customer information. The operational schema under which the PDPT will function is depicted in the following diagram:

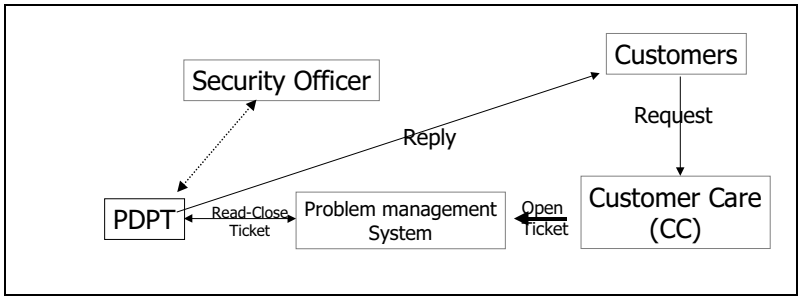


Fig.1: "Administrative Schema"

The roles and the actions of the customers, the organization's Customer Care department and the PDPT team are described below:

3.1.1. Customers:

The customers open a case that its resolution requires access to sensitive information with the organization. This service will be asynchronous in nature since the customers will be served by phone from the PDPT team after few hours or the next working day. The customers will be served in an asymmetrical way. The customers will call the Customer Care (CC) department to report the problem, but they will receive the answer from the PDPT department.

3.1.2. Customer Care:

The Customer Care department receives the requests from the customers. Having received a request, which regards sensitive information, the CC operator should open a related ticket, via the problem management system. The CC operators should not be able to close a ticket of this type.

3.1.3. PDPT

The PDPT team should have read access to the problem management system and should be able to view only the classified type of tickets. The PDPT members should not be able to open a ticket, but they should be able to close a ticket of this type. A PDPT member will undertake the job to serve the particular ticket. A security administrator will then assign the appropriate application privileges to the PDPT member. Finally, the PDPT member will communicate with the requesting customer in order to resolve the issue that corresponds to the specific ticket. After the issue has been resolved, the PDPT member will close the ticket. By doing so the closing time and the PDPT user ID should be registered in the problem management system.

3.2. Technical Approach

Having defined the possible administrative schema under which the access to the sensitive customer data of the legacy system will occur, we present in this section the logical design details of the proposed approach.

3.2.1. Content provider approach

The proprietary legacy applications are not able to support authentication authorization and accounting (AAA) functions[6]. These functions, however, are fundamental in order to ensure that the access to sensitive information will be controlled and monitored. This inherent lack of security opens the business up to potential negative consequences in the form insider misuse and leak of information. These legacy databases are integrated with other legacy subsystems in order to obtain this information from various sources and store it in a structured manner. They are, therefore, necessary components of the organization's IT infrastructure and their replacement could have major ripple effects. This is why every organization is very reluctant to fully replace them. On the other hand, there is a strong need for a secure access to the information. For that reason the sensitive information could still be gathered and stored in the legacy databases but as next step their contents could be replicated to another database platform. This platform, which we may call it content provider platform (figure 2) should be capable of performing the necessary AAA functionality. In other words, the business requirements of the PDPT services may be satisfied by the model of content provider. The transactions take place only on the legacy system whereas the platform just makes possible the view of the sensitive content when the conditions defined by the security policy can be met. Subsequently, direct access to the legacy system will be denied for all users except for the users that need to update or modify non-sensitive information. The content provider will be the only source that will reflect the sensitive information to the authorized PDPT members. The content provider platform will be implemented by a modern RDBMS system with enhanced security mechanisms and will engage mechanisms that allow data to be exported from the legacy database and imported to a modern RDBMS system. Most modern RDBMS provide utilities that could load data from external files into tables in the databases. The utilities could accept input data in a variety of formats (for example ASCII delimited files), can perform filtering, and can load data into multiple database tables during the same load session. The specific fields that should be included in this import process shall be defined by the security policy of the organization with regard to the sensitivity of the content. Based on the estimated load and bandwidth limitations, the process of periodic content replication is expected to last less

than two hours. It also will be scheduled to take place during the hours of less utilisation of the mainframe in order to avoid degradation of its performance.

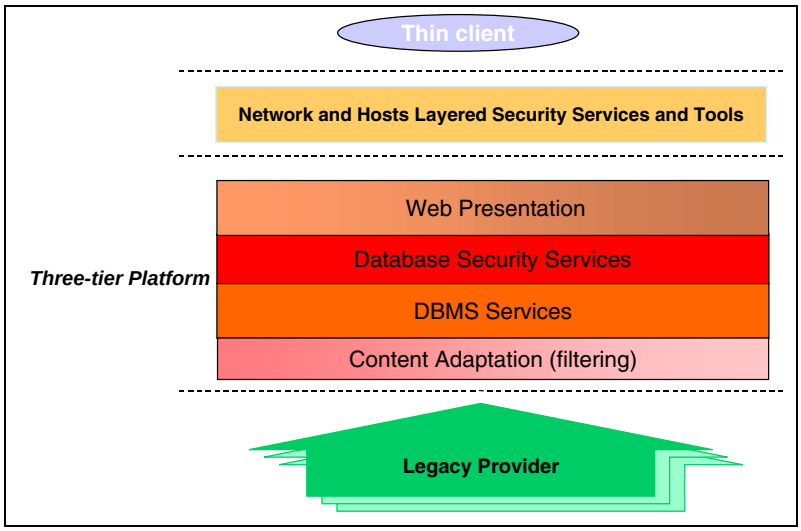


Figure 2: Content Provider Model

3.2.2. Three-tier architecture considerations

A three-tier architecture (figure 3) will be adopted for the content provider platform, in order to enhance the efficient resource management, improved scalability and security. In a three-tier system, the middle tier, typically implemented by deploying web servers, can act as a concentrator, mediating access to the back end system and allowing many user devices to share a relatively few connections to the back-end system (database server).

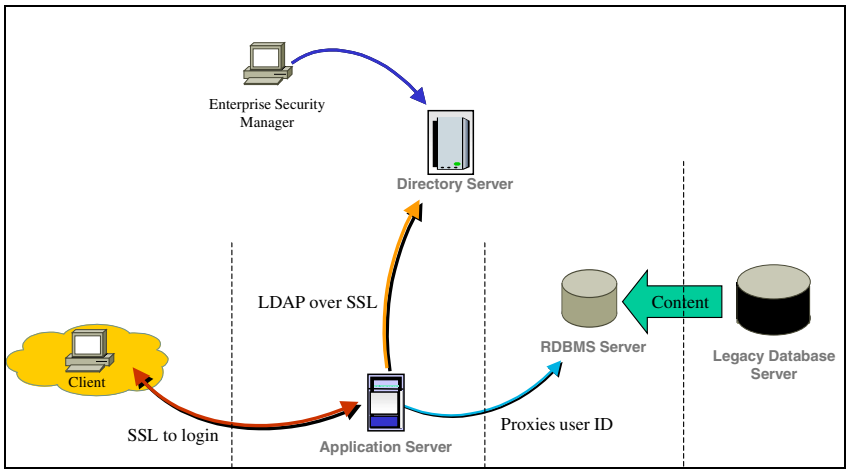


Figure 3: Three-Tier Architecture

In a three-tier system the middle tier can focus on presentation of data to the user, allowing the back end system to focus on management and processing of data, improving the platform's efficiency and scalability. Moreover, application logic in the middle tier can limit access of users, and provide another layer of isolation to sensitive data maintained in database. This improves system security. Using web servers in the middle tier allows use of thin clients (i.e., web browsers), which are easy to use and can greatly reduce overall system cost for systems which support large user communities and need to access several database systems located in various sites of the corporate network. In this respect, a three-tier content provider platform could be a model for cost-effective implementation of business services that require access to confidential data and are currently supported by legacy systems, which lack the security features of today's tools.

3.2.3. Application design considerations

The aspects of security relevant to the platform database can be implemented through appropriate authentication of users, access control to objects and auditing.

3.2.3.1. Authentication of users

It is important to ensure that all users connecting to the database authenticate. PKI functionality should be incorporated into the information exchanges involved between the clients used by PDPT members and the platform, using X.509 identity certificates instead of static passwords[3]. Typically, in the three-tier architecture, the workstation would authenticate with the application server, and the application server would authenticate with the database server. The PDPT member will be authenticated via a strong authentication mechanism to the RDBMS or to the front-end and after his/her successful authentication will be able to retrieve the required information.

3.2.3.2. Access control to objects and authentication of authorised applications

Access control to objects and management of users can be simplified through the use of roles. Roles are a collection of privileges that can be assigned to users. In a modern RDBMS, these roles can be defined to include only the operations required to complete the job function, enforcing the principle of less privilege. The user in order to be able to view sensitive information will have first to be assigned a proper database role. The role should grant the privilege to execute a stored procedure that will create the required view, update the audit table and finally revoke the privileges. By revoking the privileges, the user should effectively assume only a default role, which does not grand access to critical information. The user in order to be able again to access sensitive information will have to reassume a privileged role. Therefore, the security administrator should reassign again the proper role to the user. A dedicated person should act as security administrator will perform user privilege management by assigning the necessary roles to the PDPT operators as needed. The security administrators should also be assigned specific roles, which will allow the management of user privileges. Their activity should also be audited. The security administrators should not own any tables in the database either and should only be able to create views of the audit tables in order to generate the user activity reports.

3.2.3.3. Auditing

The content provider platform should include many features that allow the auditing of database access and operation. Auditing of the database should assist the security administrators to detect any unauthorised or malicious activities. The use of three-tier architectures creates an additional level of complexity to logging. However, the application server should pass on the identity of the real user to the database. Therefore, the user, on behalf of which the application server accesses the database information, is not masked. At a minimum, the audit table should be populated by :

- ticket number of the case the PDPT member serves
- user ID of PDPT member who viewed the information;
- what date-time the information was viewed
- what fields were selected.
- parameters needed to be provided in order to execute the SQL statement

However, logging of activities can only assist the detection process only when the logs are reviewed regularly. Logging large amount of data might create problem in the reviewing process and could have an impact to the performance requirement of the database. A summary report could then be generated from the log data to reduce the time required to review the audit information. In the next section is presented a method for the automated generation of activity reports that helps for the near real-time intrusion detection.

3.2.3.4. Automated reporting

In the framework of the content provider we can define as normal user behavior a set of well-defined events that appear in sequence. In fact this set is composed by four major events that should occur in the following time order (figure 4):

- A ticket of specific type is opened. To resolve these types of cases, access to sensitive customer personal data is required.
- A security administrator reads the ticket and assigns privileges to an PDTD member
- An operator connects to the database (through the middle tier) and access the data
- The operator closes this ticket

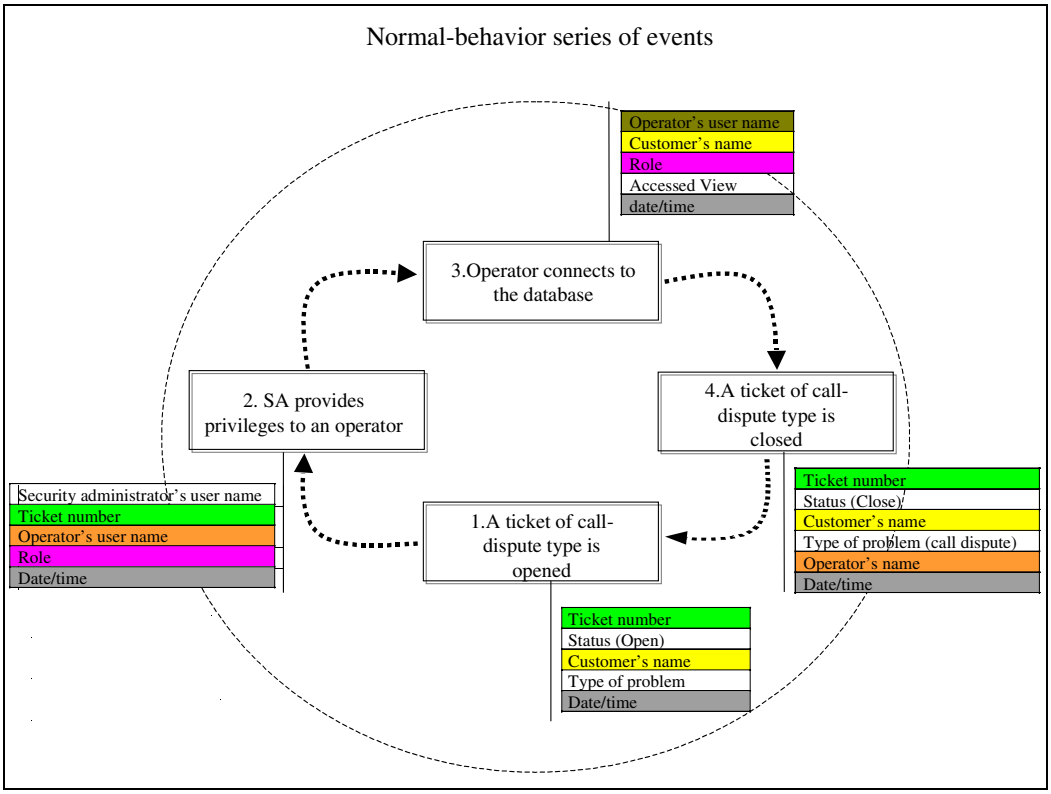


Figure 4: Normal user behavior

A ticket number uniquely identifies every ticket. When the security administrator provides the privileges, the application should require her to provide that ticket number. That number should be recorded to the audit table of the database, along with administrator's user name, date/time, operator's user name and the role that the operator was assigned. The operator's user name along with the procedure she executed, the customer's user name, date/time should also be logged in the audit table. Finally the ticket with the specific ticket number is closed. There is log information, located in two different systems that will be useful to trace and it would be helpful for incident analysis to collect these logs in a central analysis server. The central analysis server will be the center of the automated reporting operation (figure 5). This server would ideally consist of a database and a Web server. This should allow the interactive querying of log data for analysis. Also, an easy to use Web interface will help to evaluate the current attack status of the PDPT operations. It will also allow analysts to perform pre-programmed queries, such as aggregation and statistics gathering, to identify suspicious patterns and to perform rudimentary incident analysis. Finally, the information gathered in the server will provide a broader view of the user community activity in respect with the content provider platform.

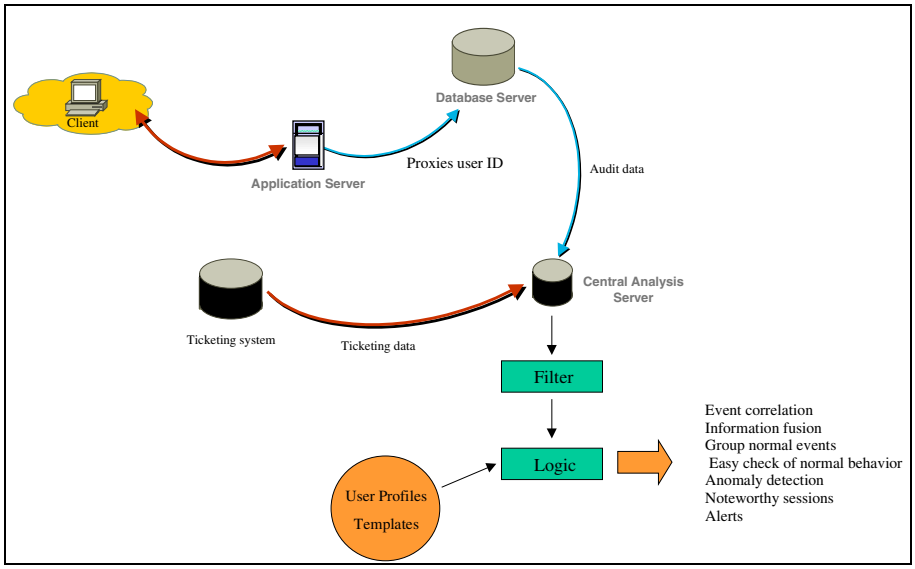


Figure 5: Automated Reporting Functionality

Event aggregation is the fundamental part of the automated reporting process. This part of the system is programming logic based on the analysis server. Aggregation simply refers to the method in which the audit information gathered from the platform is ordered.
Whenever a ticket of sensitive type is opened it should be forwarded to the central analysis server. Also a status update of such a ticket should also update the analysis server. In this manner the analysis server will include the following information for the tickets of interests:

- Ticket number
- Status (Open/Pending/Closed)
- Customer's name
- Type of problem (ticket type ID)
- (Open by) Customer Care User ID
- (Closed by) PDPT User ID
- Date/time Opened
- Date/time Closed

From content provider audit tables will be obtained logs for the security administrator activity

- Security administrator's user ID
- Ticket number
- Operator's user name
- Role
- Date/time

Also, from the content provider audit tables will be obtained logs for PDPT member activity

- PDPT User ID
- Customer's name
- Role
- Ticket number
- Executed procedure
- Date/time

From the collected data is possible to correlate the events and analyze the user community behavior. It is apparent that the date/time entries should follow a particular order. The benefits of this correlation is to simplify the security analyst's job.

3.2.4. Automated reporting

The normal behavior is represented by a quad of database events (ticket-open to ticket-close) as depicted in figure 4. These events share a common ticket number. Therefore, aggregating by ticket number, the normal events (which are expected to be the majority) can be grouped together and checked in an automatic way.

3.2.5. Anomaly detection

When the events form a quad of a common ticket ID is closed the behavior is legitimate. This fact may provide a mechanism to identify the manifestation of a breach: when some of the events do not correspond to a known quad of events and remain uncorrelated a warning may be issued. Other benefits of this technique are:

- May provide one display that integrates audit data from the databases.
- It is more intuitive, a (web) interface may reduce the mistakes.
- It lowers the degree of attentiveness required to analyze the log information scattered in the different systems.
- It helps to identify the type and severity of the security events.
- On-screen auxiliary data could be provided to understand the event.
- Facilitates the collection of events by several different attributes.
- It provides anomaly detection with a rule-based approach
- It may establish a reliable mechanism for back-tracing security breaches and assigning liabilities.

In the future, it would be useful to be able to correlate the database audit information with log information from other devices, which support directly the PDPT service, like the logs of the firewall.

3.2.6. Network based Intrusion detection

Although this paper's focus is not on network based intrusion detection, a network based IDS would increase the overall responsiveness of the content provider model securing the core system components. The IDS should be implemented on the back-end segment (where the RDBMS is reside) and be configured in a very restrictive stance (figure 6). Because the types of traffic in this particular network segment should be very specific, any signature match that may occur should be treated with an immediate response[4,5,7]. There are not anticipated any performance issues related to the IDS, since it should be configured to filter the segment's traffic against a specific subset of signatures (matching against the signatures, which are related to other TCP or UDP services are not needed because this traffic should be blocked).

3.2.7. Scalability considerations

3.2.7.1. Scaling the front-end system

In order to scale the front-end system in a three-tier architecture (figure 6), then the application server could be based on a multiprocessor system on which more memory and more processors could be added (vertical scaling). Another way to scale the front-end systems is by cloning (horizontal scaling) them in which case the hardware, software, and data are replicated.

3.2.7.2. Scaling the back-end system

Planning for growth and being able to add compute cycles when they are needed will help to avoid running out of headroom and rendering the platform unproductive. Although in general the database servers are not cloned (due to data synchronization issues of their transactional nature) in our case of the content provider platform the horizontal scaling could be feasible; for example, two database servers allow views to exactly the same content, which has been acquired by the same original source. Because, at some point it is no possible to increase the processing power and main memory, or it becomes undesirable to have so much data dependent on the availability of a single system it becomes more efficient and more economical to scale horizontally.

3.2.7.3. Network architecture Scaling

To achieve maximum scalability, the network should be built in a modular fashion. Also, for best performance the application server module should be directly attached to the Core/Distribution level of the organization's data network.

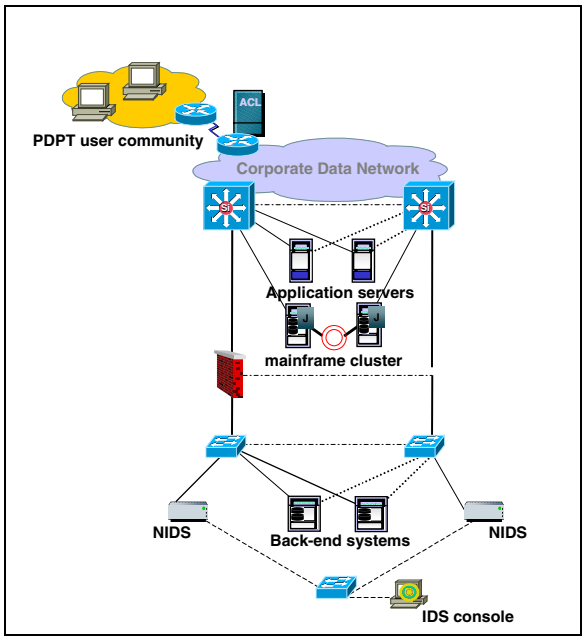


Figure 6: Scalability-Availability considerations

3.2.8. Availability considerations

3.2.8.1. High Availability from Host perspective

In order to increase the availability apart from the primary database host, a secondary database host is required. The secondary database host could be just a clone of the primary, having the same content, providing a session level load balancing and not maintaining the session state. This is also the solution with lowest cost and is simple to configure and maintain.

3.2.8.2. High Availability from Network perspective

To achieve resilience in the network infrastructure, it is recommended that the connections should be diverse; that is, communication facilities should take physically separate paths from the content provider platform to the PDPT user community. In order to avoid single point of failure and to ensure maximum accessibility to the database, high availability must be maximized at all layers of the network design. This requires two of each prerequisite network components. Within the infrastructure, switches, firewalls and routers should be interconnected in such a way that there are always multiple paths to each service and there is the ability to automatically reroute traffic across these redundant links and devices providing at least one alternative route for any failure. In other words a high availability design should ensure that layer-2 protocols such as Spanning Tree (STP) are kept in check and do not create lengthy outages during a link or device failure within a layer-2 domain. At a higher network level, high availability amongst layer-3 protocols ensures that fast rerouting is possible during IP transport failure. This is required specially in case that the PDPT team will be located in a corporate site that is connected to the content provider platform's site over WAN links. This way, no single point of failure can prevent access to the application. This also allows taking the devices off-line one at a time, minimizing planned downtime.

4. Conclusions

In this paper we have provided an overall framework for the secure integration of the mission-critical legacy systems and into modern e-business environments. The main focus of our solution was to mitigate the trusted insider misuse threat by providing both a sound from a security perspective platform that allows regulated and controlled access to sensitive corporate information and a core foundation for the near real-time detection of intrusion attempts based on a behavioral model. The overall solution that we proposed follows typical paradigms of information security practices and is based on the design of a proper security policy and administrative schema that controls the human resources necessary to implement the desired functions as well as the design of a security platform that supports the functions of the administrative schema and the security policy. The platform that we designed is based on the content provider model and utilizes a three-tier architecture in order to access the legacy mission-critical system. We have analyzed the platform in terms of Authentication, Authorization and Accounting (AAA) functions with special emphasis given on the Accounting feature of the solution. Besides the typical auditing mechanisms that every such platform should implement we propose a method for detecting near real-time intrusion attempts based on the normal behavior of the users of the system as well as the business process that the platform supports. The overall design and typical deployment scenario of such a platform has been provided to serve as a roadmap for future implementations. An analysis of the scaling and availability characteristics of the security platform has also been discussed.

References

1. The Challenges of Insider Misuse-Peter G. Neumann, http://www.csl.sri.com/~neumann/pgn-misuse.html
2. Building Internet Firewalls – Zwicky, Cooper, Chapman – O'Reilly ISBN1-56592-871-7
3. Implementing PKI Technology – Nash et al. – RSA Press ISBN 0-07-213123-3
4. Security Architecture – King et al. – RSA Press ISBN 0-07-213385-6
4. Automated Intrusion Detection Using NFR: Methods and Experiences - Wenke Lee, Christopher T. Park, and Salvatore J. Stolfo, Columbia University, USENIX Workshop on Intrusion Detection and Network Monitoring, 1999
5. Real-time Intrusion Detection and Suppression in ATM Networks R. Bettati, W. Zhao, and D. Teodor, Texas A&M University USENIX Workshop on Intrusion Detection and Network Monitoring, 1999
6. Criteria for Evaluating AAA Protocols for Network Access (RFC 2989) http://www.ietf.org/rfc/rfc2989.txt
7. Intrusion Detection – Rebecca Bace- Pearson Higher Education; ISBN: 1578701856

Building Secure Software

Dr Gary McGraw
CIGITAL
21351 Ridgetop Circle
Suite 400
Dulles, VA 20166
USA

In the interests of readability and understandability, it is RTO policy to publish PowerPoint presentations only when accompanied by supporting text. There are instances however, when the provision of such supporting text is not possible hence at the time of publishing, no accompanying text was available for the following PowerPoint presentation.

Click here to view PowerPoint presentation; Press Esc to exit

Paper presented at the RTO IST Symposium on "Real Time Intrusion Detection", held in Estoril, Portugal from 27-28 May 2002, and published in RTO-MP-101.

This page has been deliberately left blank

Page intentionnellement blanche

Click here to view PowerPoint presentation; Press Esc to exit

Performance Evaluation of Transaction-Based Anomaly Detection

Roland Büschkes
T-Mobile Deutschland GmbH
POB 300463, 53184 Bonn, Germany

Phone: +49 (228) 936-3485, Fax: +49 (228) 936-3309
Roland.Bueschkes@t-mobile.de

Tim Seipold, Ralf Wienzek
Aachen University of Technology - Department of Computer Science
Informatik 4 (Communication Systems)
52056 Aachen, Germany

Phone: +49 (241) 80-21412, Fax: +49 (241) 80-22220
{seipold, wienzek}@i4.informatik.rwth-aachen.de

Abstract

In this paper, we examine the performance of transaction-based anomaly detection and discuss the question, whether it is suited for the real-time monitoring of communication networks, from a theoretical and practical point of view. The paper shows under which circumstances it becomes possible to monitor an 100 Mbit/s communication link, involving up to three protocol layers and using a connection-oriented protocol on the transport layer.

1 Introduction

Several different intrusion detection techniques have been proposed throughout the last years. These techniques can be divided into two main categories, namely misuse and anomaly detection. Both categories have their specific advantages and disadvantages. And while the correctness and completeness of the detection are of major concern, performance is another important aspect.

In this paper, we examine the performance of a particular anomaly detection technique, transaction-based anomaly detection, and discuss its suitability for the real-time monitoring of local area networks.

The paper is divided into five parts. After this introduction we describe the two main categories of intrusion detection techniques, misuse and anomaly detection. Thereafter, section 3 introduces the concept of transaction-based anomaly detection and discusses the fundamentals of the related extended finite state machine model. The following section examines the concrete performance of this particular technique. This is done from a theoretical as well as a practical point of view. The concluding section 5 summarises the results and gives an outlook onto future work.

2 Intrusion Detection

An *intrusion detection system* (IDS) can be used to monitor the following objects within a communication network:

1. Users
2. Processes
3. Communication protocols

Paper presented at the RTO IST Symposium on "Real Time Intrusion Detection", held in Estoril, Portugal from 27-28 May 2002, and published in RTO-MP-101.

The deployed intrusion detection techniques differ mainly in the way of how these objects are monitored and how intrusion attempts are detected. And although users, processes and communication protocols are quite different in their characteristics, the deployed techniques can be divided into two main categories [Bac00]:

1. Misuse detection
2. Anomaly detection

2.1 Misuse Detection

Misuse detection tries to detect patterns of known attacks within the audit stream of a system, i.e. it identifies attacks directly. In order to do so it explicitly specifies attack patterns and monitors the audit stream for any occurrences of these patterns. This approach is based on the assumption that the intersection between normal and intrusive behaviour is not negligible [Bac00].

The different techniques belonging to this category mainly differ with respect to the way of how the attack patterns are specified and detected. Typically, expert systems, state transition modelling or special languages are used.

2.2 Anomaly Detection

Misuse detection, by explicitly describing the sequence of actions an attacker takes, is based on the specification of the undesirable or negative behaviour of users, processes and communication protocols. The dual approach is the specification of the desired or positive behaviour. Based on this normative specification of positive behaviour attacks are identified by observing deviations from the norm. Therefore, this technique is called anomaly detection. This approach is based on the assumption that the intersection between normal and intrusive behaviour is negligible or not present at all [Bac00].

Two general approaches exist to specify this positive behaviour:

1. Learning of behaviour, and
2. formal specification of behaviour.

The first approach is suited for the monitoring of objects, which show a dynamic behaviour (e.g. users). It is often based on statistical methods or learning algorithms.

The second approach, specification-based anomaly detection, is suited for the monitoring of objects, which show a static behaviour. It was first proposed by Calvin Ko [Ko96] and is based on the explicit formal specification of the object behaviour [Ko96].

3 Transaction-based Anomaly Detection

A specific specification-based technique is transaction-based anomaly detection. It is based on the assumption that each run of a communication protocol or process can be considered to represent a transaction, as it is known from database theory [HR83, Vos94].

For a database transaction the so-called ACID properties must hold:

1. Atomicity: All operations of a transaction must be completed, i.e. a transaction is treated as a single, indivisible unit.
2. Consistency: A transaction takes the system from one consistent state to another.
3. Isolation: Each transaction must be performed without interference with other transactions.
4. Durability: After a transaction has successfully been completed, all its results are saved in permanent storage.

These properties are also suited to classify anomalies and attacks related to processes and communication protocols.

3.1 Classification of Attacks

An implementation of a process or a communication protocol is usually based on a corresponding specification [Hol91]. In our model the specification and implementation are represented by deterministic finite state machines (DFSM) [HU79]. A DFSM $M = (Q, \Sigma, q_0, \delta, F)$ is an abstract machine consisting of a set of states Q (including the initial state q_0), a set of input events Σ and a partial state transition function δ. The transition function takes the current state and an input event and returns the next state. This next state is uniquely determined by a single input event. Some states may be designated as final states (the set F).

A state machine M accepts a language $L(M) \subseteq \Sigma^*$, which is defined by $L(M) = \{w \in \Sigma^* \mid \hat{\delta}(q_0,w) \in F\}$, with $\hat{\delta}$ being the extension of the state transition function δ from single input events to words w.

Both, database transactions and communication protocols, can be represented by finite state machines. By testing whether the ACID properties hold for a run of a communication protocol or process it is possible to detect anomalies and therefore potential intrusion attempts:

Theorem 1. *Classification of anomalies and attacks*

With regard to a specification DFSM $M = (Q, \Sigma, q_0, \delta, F)$ the possible anomalies and attacks concerning an implementation DFSM $M' = (Q', \Sigma', q_0, \delta', F')$ can be classified according to the transaction properties of atomicity, consistency and isolation.

A detailed proof is given in [Büs01, BNB01].

3.2 Extended Finite State Machine Model

According to the stated theorem it has to be checked whether the following properties hold for a protocol run:

1. Atomicity: The state machine reaches a final state in a limited amount of time.
2. Consistency: The state machine provides a legitimate transition for each of the monitored packets.
3. Isolation: Different state machines do not interfere.

Actually, the isolation property is not relevant for the monitoring of communication protocols. For the monitoring of the atomicity and consistency properties an extended finite state machine model is defined in [Büs01, BNB01].

A violation of the atomicity corresponds to an incomplete run of a protocol. However, during the execution of a protocol this incompleteness can only be determined through the use of corresponding timers. The timers determine after which time at latest a transition has to take place.

The consistency of a transition ensures that only valid messages are exchanged between the communicating peers. With regard to database theory three kinds of integrity constraints have to be considered [Vos94]:

1. Static integrity constraints, i.e. restrictions concerning the set of permissible states.
2. Transitional integrity constraints, i.e. restrictions concerning the set of states reachable from another state.
3. Dynamic integrity constraints, i.e. restrictions concerning the set of permissible sequences of states.

The transitional and dynamic integrity constraints obviously model dependencies between different states. Therefore, they are suited to model stateful protocols (e.g. connection-oriented protocols). The static integrity constraints are restricted to single states, without taking possible relationships to previous states into account. Therefore, they can be used for the specification of general packet properties and are especially sufficient for the specification of stateless protocols (e.g. connectionless protocols). For each kind of constraint a further distinction can be made, namely the distinction between protocol- and user-specific constraints. The former result directly from the protocol specification, the latter represent additional restrictions defined by the user.

The extended finite state machine model enhances the standard model by the introduction of timers, variables, which represent protocol data, and input tuples, which provide access to single fields of a protocol header. Arithmetic and Boolean expressions are used to represent the mentioned integrity constraints. Due to the focus of this paper we omit further details of the model and refer the interested reader to [Büs01].

3.3 Connectionless and Connection-oriented Protocols

The main characteristic of a connectionless protocol is the fact, that individual packets are semantically independent. Therefore, each packet can be interpreted independently and no state information has to be stored. A connectionless protocol can be modelled by an extended finite state machine with an initial state, a single final state and a single transition (see. Fig. 1).

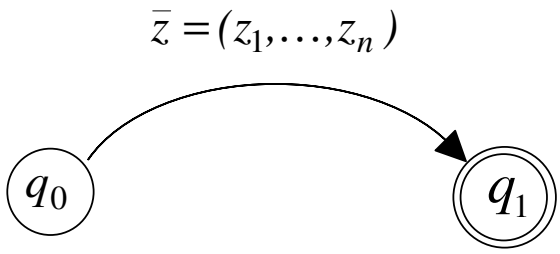


Figure 1: Modelling of a connectionless protocol.

The static integrity constraints, which have to be checked, are identical for each packet and bound to the one transition of the state machine. With regard to an element $z_i \in$ W of the input vector $\bar{z}$ the following classes of basic integrity constraints can be defined:

1. Comparison conditions, e.g. validation of the used protocol version.
2. Boolean conditions, e.g. validation of the source and destination addresses with regard to a given continuous or discontinuous address space.
3. Functional conditions, e.g. validation of the checksum.

Special kinds of Boolean conditions are the so-called range and value conditions. In the former case a given value is set into relation with an upper and lower reference value. In the latter case a given value has to be set into relation with a set of reference values.

The static integrity constraints have also to be checked in case of a connection-oriented protocol. And even the transitional and dynamic integrity constraints can be mapped onto the basic integrity constraints defined above. In addition, timers have to be administrated.

For an analysis of the complexity of operations related to the analysis of connectionless and connection-oriented protocols the reader is referred to [Büs01].

4 Performance Evaluation

While the preceding section has focussed on the general model of transaction-based anomaly detection, this section takes a closer look at the concrete performance, from both, a theoretical and practical point of view.

4.1 Model

The general objective is to monitor a typical 100 Mbit/s communication link, used for TCP/IP-based network traffic, in real-time. The monitoring should at least include the first three protocol layers, with a connection-oriented protocol being deployed on the transport layer. The following subsections define the integrity constraints, which have to be validated according to the transaction-based model.

4.1.1 DATA LINK LAYER – ETHERNET

Ethernet is a connectionless and therefore stateless protocol [Tan96]. Due to its stateless nature, the following static integrity constraints have to be validated for an Ethernet frame (see Fig. 2):

1. Protocol-specific integrity constraints
   a. Validation of the preamble field and the Start-of-frame delimiter[1] (comparison condition).
   b. Validation of the payload type[2] (comparison condition).
   c. Validation of the frame length (comparison condition).
   d. Validation of the used padding bits (functional condition).
   e. Validation of the checksum (functional condition).

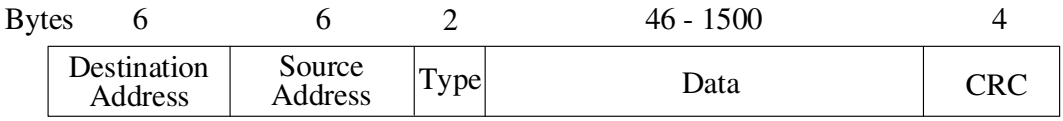


Figure 2: Ethernet frame.

In addition, the following user-specific constraints are validated:

2. User-specific integrity constraints
   a. Validation of the source and destination addresses with regard to a given list of addresses (Boolean condition).

4.1.2 NETWORK LAYER – IP

IP [Pos81a] is also a connectionless and therefore stateless protocol. Hence, only the following static integrity constraints have to be validated for an IP datagram (see Fig. 3):

1. Protocol-specific integrity constraints
   a. Validation of the protocol version (comparison condition).
   b. Validation of the datagram length (comparison condition).
   c. Validation of the header checksum (functional condition).
   d. Validation of the options (functional condition).

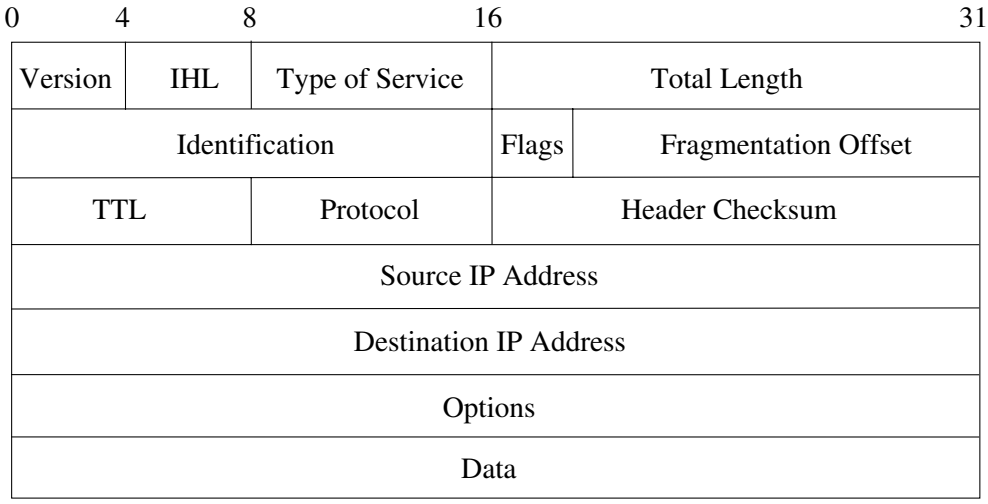


Figure 3: IP datagram.

[1] Must be supported by the network interface card.
[2] Only IP is considered on the network layer.

In addition, the following user-specific constraints are validated:

2. User-specific integrity constraints
   a. Validation of the requested service (comparison condition).
   b. Validation of the time-to-live value with regard to the source address (Boolean condition).
   c. Validation of the protocol field with regard to a given list of admissible protocols (Boolean condition).
   d. Validation of the source and destination addresses (Boolean condition).
   e. Validation of the source and destination addresses with regard to the used Ethernet addresses (Boolean condition).

4.1.3 TRANSPORT LAYER – TCP

As for all connection-oriented protocols the transmission of data by means of TCP [Pos81b] can be divided into three phases:

1. Connection establishment
2. Data transmission
3. Connection termination

During all of these three phases the following static integrity constraints have to be validated for a TCP segment (see Fig. 4):

- Static Constraints

1. Protocol-specific integrity constraints
   a. Validation of the TCP checksum (functional condition).
   b. Validation of the URG flag and the related urgent pointer (Boolean condition).
   c. Validation of the six reserved bits (comparison condition).
   d. Validation of the options (functional condition).

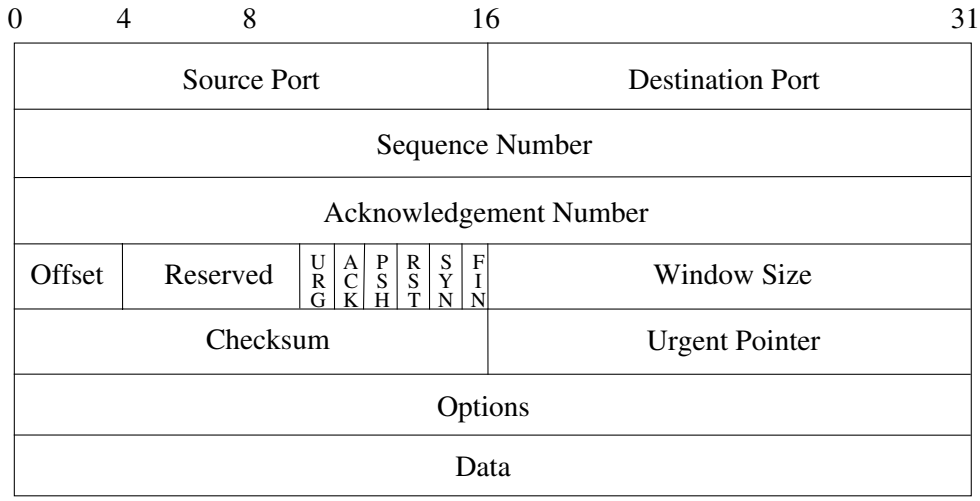


Figure 4: TCP segment.

During the connection establishment phase the following additional integrity constraints have to be validated:

- Static constraints:

1. Protocol-specific integrity constraints
   a. Validation whether the given combination of source/destination addresses and ports is not already in use (connection administration).

2. User-specific integrity constraints
   a. Validation of the given combination of source/destination addresses and ports (Boolean condition).
   b. Validation of the initial sequence number (comparison condition).

- Dynamic constraints:

1. Protocol-specific integrity constraints
   a. Validation of the correct execution of the connection establishment protocol (three-way handshake, 3WHS), which is represented by a corresponding extended finite state machine.

During the data transmission the following additional dynamic integrity constraints have to be validated:

- Dynamic constraints:

1. Protocol-specific integrity constraints
   a. Validation whether the segment belongs to an existing connection (connection administration).
   b. Validation of the used acknowledgement number with regard to the actual sequence number and window size (comparison condition).
   c. Validation of the sequence number (Boolean condition).

Actually, additional constraints result from the TCP error correction mechanisms. However, an ideal, error-free connection is assumed in order to simplify the discussion.

During the connection termination phase the following additional static and dynamic integrity constraints have to be validated:

- Static constraints:

1. Protocol-specific integrity constraints
   a. Validation whether all data has been transmitted before terminating the connection (comparison condition).

- Dynamic constraints:

1. Protocol-specific integrity constraints
   a. Validation whether the segment belongs to an existing connection (connection administration).
   b. Validation of the correct execution of the connection termination protocol, which is represented by a corresponding extended finite state machine.

4.2 Parameters

To parameterise the given model it is necessary to determine the time required for the basic operations. Therefore, corresponding measurements were made on a standard PC architecture (400 MHz, 256 MB) using 32 bit integer numbers.

Two basic comparison operations were considered. As an example for a simple operation, which can be mapped onto hardware, the < relation was chosen. In addition, the more complex $\circ$ relation was constructed as follows: $a \circ b \Leftrightarrow (a + b) \text{ MOD } M = R$ for an arbitrary chosen $M \in \{0,...,2^{32}-1\}$ and a reference value $R < M$.

The performance measurements showed an average duration of $t_0^{<}$ = 12.98 ns and t_0° = 136.6 ns respectively, i.e. the hardware supported comparison operation < can be executed 10 times faster than the more complex $\circ$ operation.

The time need for the evaluation of range and value conditions can directly be derived from these results. A functional condition is more complex. It is composed of a function calculation over a set of parameters and the comparison of the calculated value with a given value. As an example for a non-trivial function the MD5 algorithm is chosen, for which, with regard to the used hardware platform and in accordance with [Tou95], an average throughput of 182 Mbit/s can be assumed. This corresponds to t_1^{MD5} = 43.96 ns/Byte.

Besides these operations for the evaluation of integrity constraints, several tree-based functions are required for the administration of connection-oriented protocol instances and timers.

Based on corresponding measurements and assuming a generic function of the form $f(x)=a*\ln(b*x)$ the function $T(x)= 162.2*\ln(58.68*x)$ ns was determined for the used hardware platform and a standard implementation.

Given these values the model can be analysed. With R_i^j denoting the execution time of step i on layer j the total time need is $R = \sum_j \sum_i R_i^j$.

4.2.1 DATA LINK LAYER – ETHERNET

The time necessary for the validation of the constraints defined in section 4.1.1 can be estimated as follows:

1. 3 comparisons for the validation of the preamble, the Start-of-frame delimiter and the frame length: $R_1^1 = 3*t_0$.
2. Validation of the checksum for a packet of length p_1: $R_2^1 = p_1*t_1$.

This results in a time need of:

$$R^1 = \sum_i R_i^1 = 3*t_0 + p_1*t_1$$

Compared to the constraints defined in section 4.1.1 the validation of the addresses, payload type and padding data is omitted. The number of comparisons necessary for the validation of these and other constraints will be estimated later on.

4.2.2 NETWORK LAYER – IP

Based on the constraints defined in section 4.1.2 the time necessary for the validation of an IP datagram can be estimated as follows:

1. Validation of the protocol version, the given packet length and the service: $R_1^2 = 3*t_0$.
2. Validation of the source and destination addresses under consideration of address dependencies defined by a Boolean expressions of length p_2: $R_2^2 = p_2*t_0$. Alternatively, if no dependencies are defined, validation of a range condition on the basis of the address and a given network mask: $R_{2'}^2 = 2*(t_0+t_1)$, i.e. a binary AND operation and a comparison each.
3. Validation of the time-to-live value in dependency on the source address[3]: $R_3^2 = 6*t_0$.
4. Validation of the header checksum assuming the maximum header length: $R_4^2 = 60*t_1$. Alternatively, validation without option fields: $R_{4'}^2 = 20*t_1$.
5. Validation of the options assuming the maximum header length: $R_5^2 = 40*t_1$. Alternatively, no options to be validated: $R_{5'}^2 = 0$ ns.
6. Validation of the source and destination addresses on the IP and Ethernet layer: $R_6^2 = 4*t_0$.
7. Validation of the protocol field on the basis of a given list of admissible protocols with p_3 entries: $R_7^2 = p_3*t_0$.

[3] The validation is restricted to the differentiation of internal and external addresses (based on the given address and a specified network mask) and the related time-to-live values.

Therefore, in the worst case, the time required for the validation of an IP datagram is:

$$R^2 = \sum_i R_i^2 = (p_2 + p_3 + 13) * t_0 + 100 * t_1 \text{ with } i \in \{1,...,7\}.$$

In the average case, the needed time can be reduced to:

$$R^{2'} = \sum_i R_i^2 = (p_3 + 15) * t_0 + 22 * t_1 \text{ with } i \in \{1,2',3,4',5',6,7\}.$$

4.2.3 TRANSPORT LAYER – TCP

The time required for the validations, which have to be done in all three phases of a TCP connection, can be estimated as follows:

1. Validation of the checksum in dependency on the segment length p_4: $R_1^3 = 1 * t_0 + (p_4+12) * t_1$.
2. Validation of the URG pointer and the reserved bits: $R_2^3 = 3 * t_0$.
3. Validation of the options. Due to the many possible kinds of usage, the required time is estimated by two function calculations per option word: $R_3^3 = 11 * 2 * t_1$.
4. Association of the packet with one of the existing p_5 TCP connections and, in parallel, determination of the state of the corresponding extended finite state machine and the related variable assignment, or alternatively initialisation or deletion of this information: $R_4^3 = T(p_5)$.

Considering the different phases of a connection, it has to be taken into account that the time required per segment has to be determined, i.e. the time required for a single transition of the corresponding state machine. As the time needed for the individual transitions differs, the worst-case has to be analysed.

For the connection establishment phase the following times have to be considered:

1. Validation of the source and destination ports and their dependencies (Boolean condition of length p_6) and validation of the corresponding header fields for connection establishment: $R_1^{3'} = (p_6+3) * t_0$. In general, this test is only necessary for the first packet of a connection.
2. Verification of the connection establishment process (3WHS). In order to do so an arithmetic expression of length 2 has to be calculated and four values have to be compared: $R_2^{3'} = 4 * t_0 + 2 * t_1$.

For the data transmission phase the following time has to be considered:

1. Determination of the transition of the state machine by evaluating an arithmetic expression of length 2 and comparing 4 values: $R_1^{3''} = 4 * t_0 + 2 * t_1$.

For the connection termination phase the following time is required:

1. Determination of the transition of the state machine: $R_1^{3'''} = 5 * t_0 + 2 * t_1$.

In order to determine the total amount of time needed for the validation of a single segment, all three phases have to be taken into account. Because a separated handling for each phase would unnecessarily complicate the analysis, the time is estimated by combining the worst-cases for each of the three components t_0, t_1 and T(...) for each of the three phases, resulting in $R_5^3 = (p_6+3) * t_0 + 2 * t_1$ with $p_6 \geq 2$. This results in a total time need of:

$$R^3 = \sum_i R_i^3 = (p_6 + 7) * t_0 + (p_4 + 36) * t_1 + T(p_5)$$

4.3 Theoretical Performance

In total the analysis of a TCP packet requires the time shown in Tab. 1.

The variables p_1 and p_4 are related and can be summarised into a single variable, representing the payload of an Ethernet frame. The variables p_2, p_3 and p_6 are configuration-dependent factors of t_0. Therefore, they can also be summarised into a single variable, which represents the number of all comparison operations needed for the validation. Although the variable can take any value, only a value range of {1,...,100} is considered to be of practical interest. For the following analysis, the variable is set to its maximum (100).

Layer	Protocol	Time R^i in ns
3	TCP	$(p_6+7)*t_0 + (p_4+36)*t_1 + T(p_5)$
2	IP[4]	$(p_2+p_3+13)*t_0 + 100*t_1$
1	Ethernet	$3*t_0 + p_1*t_1$
Time		$(p_2+p_3+p_6+23)*t_0 + (p_1+p_4+136)*t_1 + T(p_5)$

Table 1: Total time need.

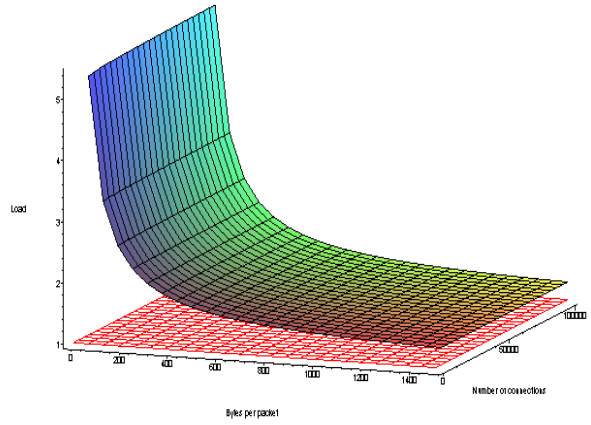
Fig. 5(a) shows the resulting load for $t_0 = t_0^{\circ}$ and $t_1 = t_1^{MD5}$ with regard to an 100 Mbit/s communication link. Obviously, the load is always higher than 1, for small packets even by a factor of 5. However, a variation of the number of active connections does not have a significant influence on the load.

Modifying the scenario and considering $R^{2'}$ instead of R^2, $t_0 = t_0^{<}$ and $t_1 = t_1^{MD5}/2$ results in the load shown in Fig. 5(b). Although the load is lowered, it is still above 1 for small packets. With 50000 active connections the load for packets with less than 96 Bytes raises above 1 and reaches in the worst case a value of 1.3.

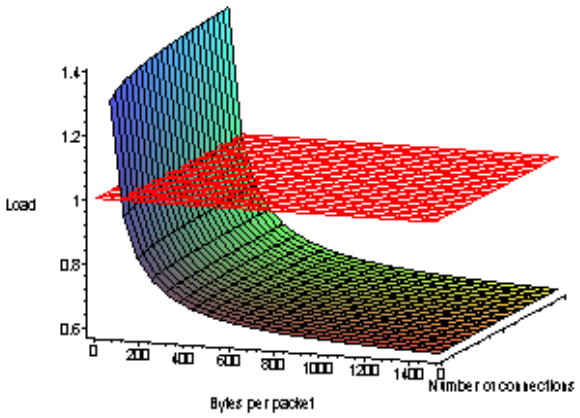
For a further optimisation the influence of the function T(...) has to be analysed. Fig. 5(c) shows the variation of the load in dependence of the run-time of the function T(...) and a fixed number of connections (100000). The load drops below 1 only after reducing the original value by 85%. Therefore, in Fig. 5(d) the number of tests is lowered to 25. But even in this case the time required for the connection administration has to be reduced by 50% in order to guarantee a reliable monitoring of all TCP connections. Correspondingly, Fig. 5(e) and 5(f) show the load behaviour for T(...)/2 and a fixed number of 100 and 25 tests respectively. Only in the last case the load is permanently below 1. For this configuration it becomes possible to reliably monitor an 100 Mbit/s communication link by means of transaction-based anomaly detection.

For the analysis of the resulting memory requirements and other TCP/IP-based scenarios including the application layer the reader is referred to [Büs01].

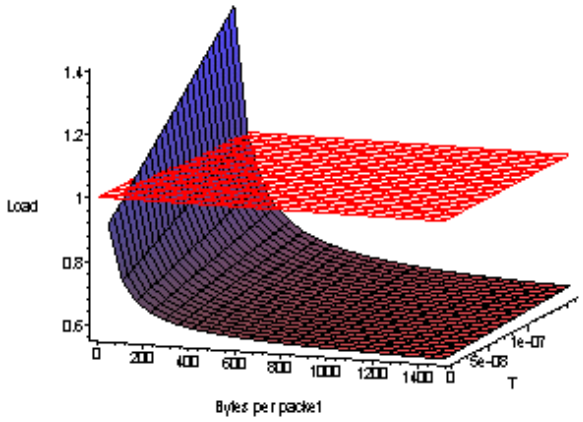
[4] Alternatively, $(p_3+15)*t_0+22*t_1$.



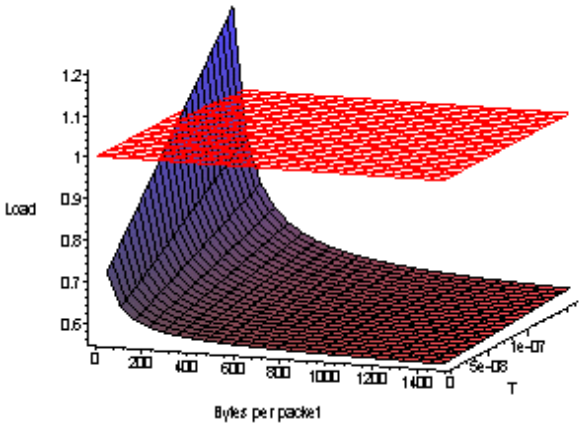
(a) Scenario Analysis ($t_0 = t_0^{\circ}$, $t_1 = t_1^{MD5}$, T(...), R^2)



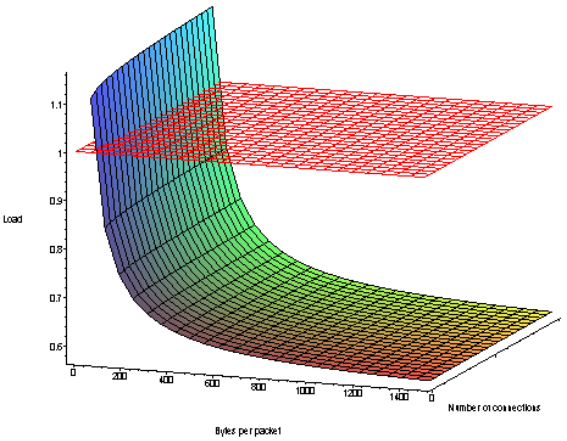
(b) Scenario Analysis ($t_0 = t_0^{<}$, $t_1 = t_1^{MD5}/2$, T(...), $R^{2'}$)



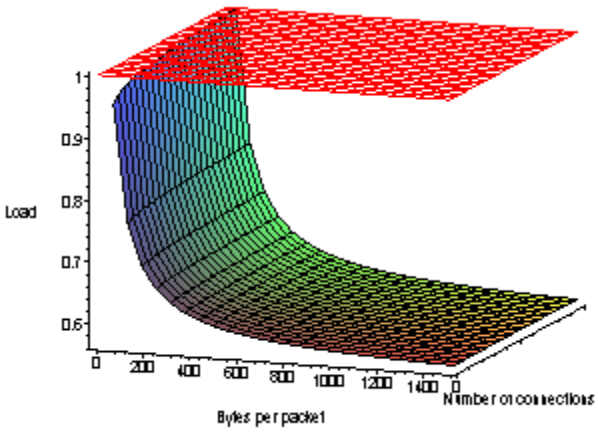
(c) Dependence on T(...) with 100 tests ($t_0 = t_0^{<}$, $t_1 = t_1^{MD5}/2$, $R^{2'}$)



(d) Dependence on T(...) with 25 tests ($t_0 = t_0^{<}$, $t_1 = t_1^{MD5}/2$, $R^{2'}$)



(e) Scenario analysis with 100 tests ($t_0 = t_0^{<}$, $t_1 = t_1^{MD5}/2$, T(...)/2, $R^{2'}$)



(f) Scenario analysis with 25 tests ($t_0 = t_0^{<}$, $t_1 = t_1^{MD5}/2$, T(...)/2, $R^{2'}$)

Figure 5: Theoretical scenario analysis (Ethernet, IP and TCP).

4.4 Practical Performance

For the practical performance evaluation of transaction-based anomaly detection a corresponding IDS prototype has been implemented. The prototype is based on Microsoft's COM/DCOM technology [Box98].

The performance measurements were done on a standard PC architecture (500 MHz, 128 MB). Fig. 6 shows the total throughput for a TCP connection, which is monitored according to the integrity constraints defined above. Considering all three phases of a connection (Total), i.e. connection establishment, transfer of a single packet and connection termination, a throughput of only 27 Mbit/s is reached in the best case. Considering only the data transmission phase (Data) a total throughput of 118 Mbit/s is reached. However, in the worst case this value drops to 0.14 Mbit/s.

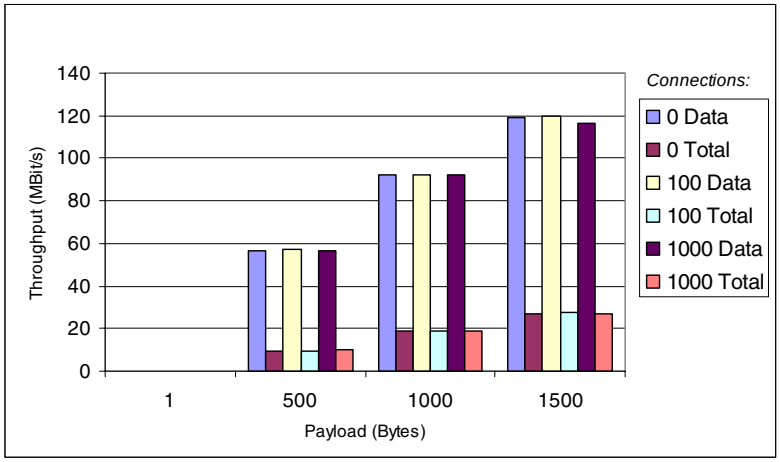


Figure 6: Practical scenario analysis.

In general, the practical performance measurements confirm the analytical results and especially the negative influence of small packets, which can potentially be misused by an attacker. The measurements show also that the prototype must be further optimised in order to get near to the results of the theoretical performance evaluation.

Concerning the practical performance evaluation of additional TCP/IP-based scenarios the reader is again referred to [Büs01].

5 Conclusions

In this paper we have discussed the theoretical and practical performance of transaction-based anomaly detection. The theoretical performance evaluation has shown that a network link providing a bandwidth of 100 Mbit/s can be monitored by an IDS deploying this special kind of anomaly detection. However, several optimisations are necessary in order to do so. The preliminary performance results of the developed prototype show the additional complexity added by the used operating system and middleware platform, which results in a corresponding performance loss. Although the software prototype does have sufficient potential for optimisation, special hardware implementations should be seriously considered for any large scale deployment.

References

[Bac00] R. G. Bace. *Intrusion Detection*. Macmillan Technical Publishing, 2000.

[BNB01] R. Büschkes, T. Noll, and M. Borning. Transaction-based anomaly detection in communication networks. In *Proceedings of the 9th International Conference on Telecommunication Systems, Modelling and Analysis*, pages 33-47, March 2001.

[Box98] D. Box. *Essential COM*. Addison Wesley, 1998.

[Büs01] R. Büschkes. *Angriffserkennung in Kommunikationsnetzen*. PhD thesis, Aachen University of Technology, May 2001.

[Hol91] G. J. Holzmann. *Design and Validation of Computer Protocols*. Prentice-Hall, 1991.

[HR83] T. Härder and A. Reuter. Principles of transaction-oriented database recovery. *Computing surveys*, 15(4):287-317, 1983.

[HU79] J. Hopcroft and J. Ullman. *Introduction to Automata Theory, Languages, and Computation*. Addison-Wesley, 1979.

[Ko96] C. C. W. Ko. *Execution Monitoring of Security-Critical Programs in a Distributed System: A Specification-Based Approach*. PhD thesis, University of California, Davis, 1996.

[Pos81a] J. Postel. *RFC 791: Internet Protocol*. September 1981.

[Pos81b] J. Postel. *RFC 793: Transmission Control Protocol*. September 1981.

[Tan96] A. S. Tanenbaum. *Computer Networks*. Prentice Hall Press, 1996.

[Tou95] J. D. Touch. Performance analysis of MD5. In *Proceedings of ACM SIGCOMM Conference 1995*, Cambridge, MA, USA, pages 77-86, October 1995.

[Vos94] G. Vossen. *Datenmodelle, Datenbanksprachen und Datenbank-Management-Systeme*. Addison-Wesley, 2nd Edition, 1994.

This page has been deliberately left blank

Page intentionnellement blanche

Click here to view PowerPoint presentation; Press Esc to exit

An Investigation of the Practical Limitations of Network-Based Intrusion Detection Imposed by Partial IP Datagram Inspection

Donald Macleod, David Whyte
Office of Critical Infrastructure Protection and Emergency Preparedness
122 Bank St., 2nd Floor
Jackson Building
Ottawa, Ontario K1A 0W6
Canada

Donald.Macleod@ocipep.gc.ca
David.Whyte@ocipep.gc.ca

© Minister of National Defence, on behalf of The Office of Critical Infrastructure Protection and Emergency Preparedness, Canada, 2002.

In consideration for this symposium, we submit the following paper. This paper should not be taken or interpreted as representing the views of our employer, the Department of National Defence / The Office of Critical Infrastructure Protection and Emergency Preparedness, nor those of the Government of Canada.

Abstract

The use of signature-based network Intrusion Detection Systems (IDS) are an integral part of a layered network security posture because they provide the ability to rapidly detect malicious network attacks. Signature-based network IDSs work by inspecting and comparing Internet Protocol (IP) datagrams against known network intrusion signatures. IP datagrams are composed of two basic components: (1) a header portion that contains the necessary information to route the datagram and (2) a payload portion that contains the data that traverses the network. In large networks, it is not uncommon for gigabits of IP datagrams to move across a network in seconds. This throughput presents technical challenges because an IDS must collect, inspect, analyse, store, and alarm on the IP datagrams as they move through a network.

It may not be possible to inspect an entire IP datagram for reasons such as: legal issues, privacy concerns, storage limitations, high-speed lines, and encryption. One possible solution is to perform partial IP datagram inspection. This technique enables IDSs to ignore all or a portion of the user information contained in the IP datagram payload. The purpose of this paper is to determine the impact of partial IP datagram inspection on signature-based network IDSs. The number of IDS signatures that require data payload inspection can be determined by restricting the amount of an IP datagram available for examination and then determining which current network attacks are no longer detected. These results will reveal the relationship between IP datagram capture lengths and the ability of IDSs to detect current network attacks. By reducing the amount of the IP datagram that needs to be inspected and analysed, signature-based network IDSs will be able to increase throughput and minimize the amount of incident data storage.

Paper presented at the RTO IST Symposium on "Real Time Intrusion Detection", held in Estoril, Portugal from 27-28 May 2002, and published in RTO-MP-101.

1 . Introduction

The Internet is a complex and heterogeneous environment that enables information to flow between networks almost instantaneously. It is an inexpensive, efficient, reliable, and rapid communications medium. Internet usage is growing at an exponential rate as organizations, governments, and citizens continue to increase their reliance on this technology. Many sectors of society use the Internet to deliver essential services such as: utilities, financial, government, health, and transportation. The infrastructure that supports these essential services is known as the critical information infrastructure. Safeguarding the IT systems that control the critical information infrastructure requires an effective layered network security posture.

> "Layered network security involves the development of sound organizational security policies and the strategic deployment of appropriate risk-based security measures thereby reducing the possibility of circumvention through single points of failure [1]."

Intrusion detection systems (IDS) are an integral part of any layered network security strategy.

> "Intrusion detection is the process of detecting unauthorized use of, or attack upon, a computer or network. [2]"

The strategic deployment of IDS is critical to the timely detection of malicious events on a network or an individual system. These devices determine whether an intrusion has occurred by collecting and inspecting either low-level network data or high-level system audit data. Our paper seeks to determine the impact on various network-based IDSs when their IP datagram inspection is limited due to: policy, legal, or technical reasons. One method to alleviate the concern of private information disclosure, while undertaking intrusion detection, is to perform partial datagram inspection. There are several benefits to this type of approach. Partial datagram inspection can increase the performance of IDS through reduced resource consumption (e.g. disk space, CPU usage). By minimizing the interception of user data, privacy can be protected. However, restricting the amount of the datagram for IDS inspection may impede its ability to detect network intrusions. Most commercially available signature-based IDSs contain proprietary signatures. Therefore, the signature algorithms used to detect intrusions are not available for analysis [3]. Determining the amount of an IP datagram that is required to detect specific intrusions is problematic. The amount of datagram inspection required varies. It depends on a number of factors such as: signature, type of attack, fragmentation, and protocol. In order to answer this question, five network intrusions were selected and executed in a test network. The resulting network traffic was captured, replayed across the network, and monitored by three IDSs. Each replay of an intrusion contained progressively smaller IP datagram sizes. This revealed the effect of partial IP datagram inspection on the ability of IDSs to detect intrusions.

The remainder of the paper is organized as follows: Section 2 provides background on the TCP/IP protocol suite and IDS, Section 3 describes the testing methodology and limitations caused by its assumptions, Section 4 presents our results, in Section 5 related work is covered, and finally in Section 6 we summarize our findings.

2 . Background

2.1. TCP/IP

TCP/IP is not a single protocol, but rather a suite of protocols. Using a suite of protocols simplifies the design and implementation of the hardware and software that allow computing platforms to be connected. It is comprised of [4]:

- Internet Protocol (IP): IP is a network layer protocol that is used to deliver datagrams across connected networks to their intended destination. IP is connectionless and thus datagrams may arrive out of sequence, be delayed, duplicated, or not arrive at all.

- Transmission Control Protocol (TCP): TCP provides reliable connection-based data stream delivery that ensures error-free, sequential, and non-duplicated datagram delivery. This is achieved by the use of a number of techniques such as: flow control mechanisms, sequence numbers, required acknowledgements, and adaptive retransmission.
- User Datagram Protocol (UDP): UDP provides unreliable datagram delivery. Datagrams may arrive out of sequence, be delayed, duplicated, or not arrive at all. UDP depends upon the IP protocol to move the datagrams that it produces.
- Internet Control Message Protocol (ICMP): ICMP is used to send messages between computing systems for diagnostic or management purposes.

For a detailed explanation of the exact format and description of the TCP/IP protocol suite, please refer to RFC 791 [5] and RFC 793 [6]. The TCP/IP protocol suite governs how data is transported across networks from host to host, but does not specify how data is transmitted across different physical media. To transmit these frames over physical media, most networks use Ethernet. Ethernet is an IEEE 802.3 series standard that specifies how two or more systems sharing a common cabling system can interact [7]. Ethernet uses its own addressing scheme that consists of a unique 48-bit number known as the Media Access Controller (MAC) address. The MAC address encapsulates datagrams transmitted over networks that use Ethernet.

Figure 1: Datagram Structure, illustrates the anatomy of a typical IP datagram that traverses over Ethernet [4]. Network applications generate datagrams by encapsulating data with protocol information. The protocol information appended to the data field is called the datagram header. The data portion is first encapsulated by the desired transport protocol, then by the IP protocol, and finally by the MAC addresses. To inspect a TCP datagram header, the first 54-bytes would have to be examined (14-byte Ethernet header, 20-byte IP header, and 20-byte TCP header). To inspect a UDP datagram header, only 42-bytes would need to be examined (14-byte Ethernet header, 20-byte IP header, and an 8-byte UDP header).

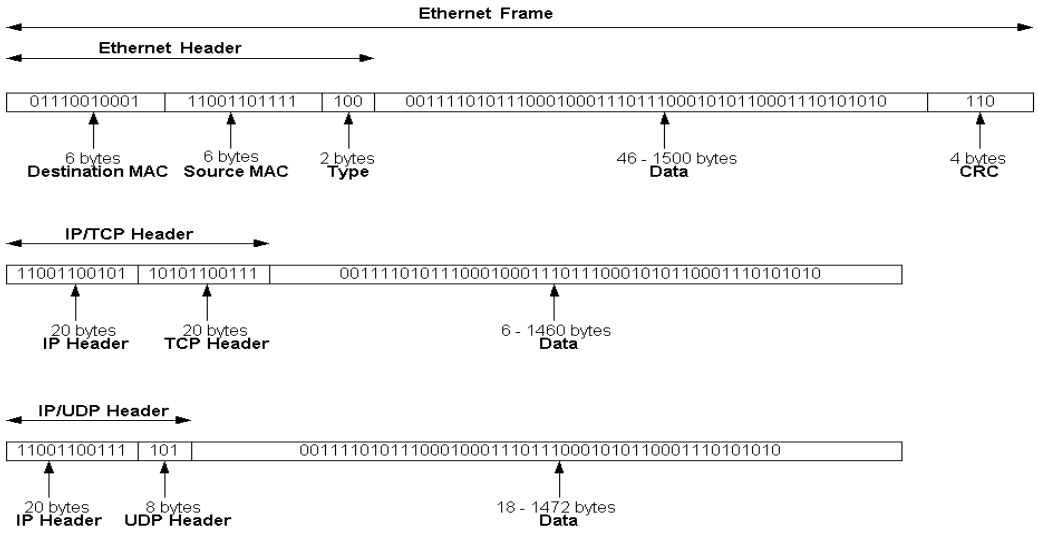


Figure 1: Datagram Structure

2.2. IDS

IDSs are an integral part of any layered network security solution. There are three main categories of IDSs: host, application, and network [2]. Host-based IDSs monitor a specific system for suspicious connections and then log the events as alarms. IDS software is installed on the system and alarms are sent to an IDS console for viewing. This type of IDS typically utilizes the pre-existing audit subsystem to generate alarms.

Application-based IDS is similar to host-based IDS. However, application-based IDS is focused on a specific application or subsystem. If suspicious activity is detected, it is logged as an alarm and sent to an IDS console for viewing.

A network-based IDS monitors a network segment for malicious activity. IDS software is installed on a system called a sensor and is connected to the network being monitored. There are two types of network-based IDSs: anomaly detection and signature detection. Anomaly-based IDSs rely on identifying unusual behaviour on a network. Any deviation from normal network behaviour is detected and then identified as a possible intrusion. A signature-based network IDS contains a database of known network attacks. Signature-based IDSs use their database of signatures to compare against the network traffic it is monitoring. If the network traffic matches an attack signature, an alarm is generated and sent to an IDS console for viewing.

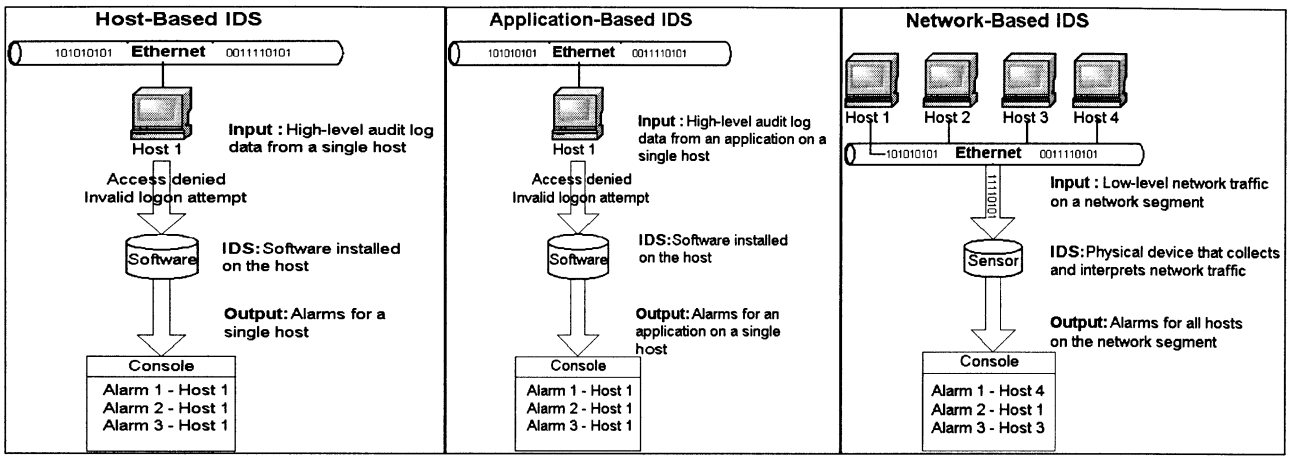


Figure 2: Categories of IDSs

Network-based IDSs monitor and inspect network traffic in order to detect suspicious activity. The deeper into the datagrams an IDS has to inspect the greater:

- the CPU resources that must be expended,
- the probability that other datagrams will be missed,
- the disk space that will be used by the storage device, and
- the likelihood that the content of the datagram could contain personal or private information.

3 . Basic Methodology

Three commercially available IDSs were selected for this study. The specific IDSs chosen are excluded from the results. This study was not intended to be comparative analyses of IDSs. Our intent is to determine the impact of partial datagram inspection on currently available network-based IDSs. The methodologies for testing IDSs are somewhat controversial. Most of these differing approaches relate to: network traffic generation, performance metrics (e.g. throughput, resource consumption), and the use of vulnerability scanners [8]. Our testing excluded all of these techniques.

3.1. Attack Tool Selection

The five network attacks chosen for this study are described in Table 1: Network Attack Tools [9].

Intrusion	Intrusion Type	Description
A. Nmap	Scanning	An open source network-mapping tool that allows a remote user to perform network exploration and security auditing.
B. Targa3	Denial of Service	An exploit-generating tool that sends combinations of invalid and unexpected IP datagrams to a victim. Datagrams may contain invalid protocol, fragmentation, packet size, flag combinations, offsets, routing flags, and other values.
C. Jill.c	Buffer Overflow – Remote Root Compromise	A buffer overflow in the printer service for IIS 5.0 that allows a remote user to execute arbitrary code on a vulnerable system.
D. Back Orifice	Trojan Horse	A program that allows a remote user to control most parts of a victim's operating system.
E. Bind8x.c	Remote Root Access	A program that allows a remote user to execute arbitrary code on a vulnerable system.

Table 1: Network Attack Tools

The attacks chosen represent a variety of threats on the Internet including: scanning, remote penetration, trojan horse, and denial of service tools. Collectively, they employ the most common protocols on the Internet: TCP, UDP, and ICMP. These tools allowed us to reveal how partial datagram inspection of common protocols would affect the ability of an IDS to detect an intrusion. The network attack tools were installed and target both the Linux and Windows operating systems.

3.2. Test Network

The test network topology is depicted in Figure 3: Test Network Topology.

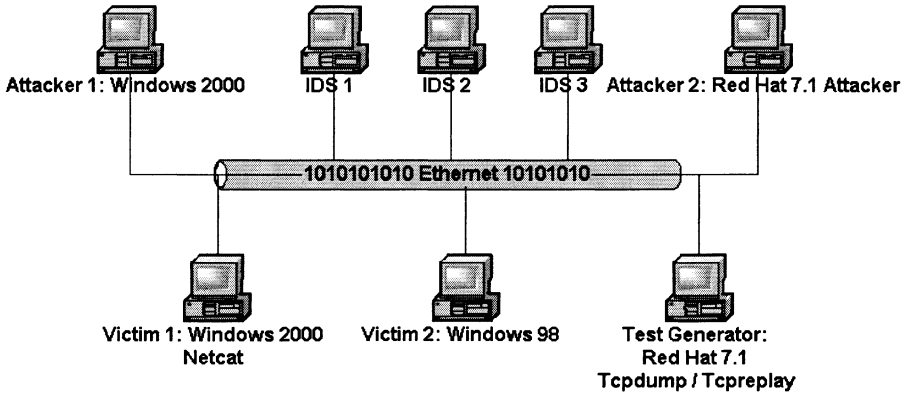


Figure 3: Test Network Topology

Five PCs and three IDSs were connected together by a hub to form a small network. The two PCs labelled "Attacker 1" and "Attacker 2" were used to execute the attack tools. The PC labelled "Test Generator" was used to capture and replay the network traffic generated by the attack tools. "IDS – 1", "IDS – 2", and "IDS – 3" were the subjects of our testing. Each IDS had the latest signature set installed and all signatures were enabled. The specific configurations of each network device can be found in Table 2: Test Network Configuration.

Test Network Configuration	
Attack Computers	Attacker 1: Windows 2000
	Attacker 2: Red Hat 7.1
Target Computers	Target 1: Windows 98, Netcat
	Target 2: Windows 2000, Netcat
Network Traffic Replayer	Test Generator: Red Hat 7.1, Tcpdump, Tcpreplay
IDS	IDS – 1: Latest signatures installed, full signature set enabled
	IDS – 2: Latest signatures installed, full signature set enabled
	IDS – 3: Latest signatures installed, full signature set enabled

Table 2: Test Network Configuration

In order to successfully execute attacks C and E, the attacking computer must make a connection with a specific port on the target system. A program called NetCat [10] was used to install a listening process on a port of the target to simulate the required open port.

3.3. Testing

3.3.1. Control Tests

Control tests were performed for each IDS to confirm that: (1) the IDSs had the requisite signatures to detect the five test attacks and (2) the intrusions would be detected under our test conditions. The first control test consisted of launching the attack tools against the victims to confirm that the IDSs would detect the attacks. Every attack was detected by all three IDSs. This confirmed the IDSs contained the required intrusion signatures and that the IDSs were able to detect the attacks in the test network.

After the first control test was completed, the attacks were executed again and then the resulting network traffic was captured using a program called Tcpdump [11]. Tcpdump has a snaplen variable that is used to specify, in bytes, the depth of datagram capture. Recall from Figure 1: Datagram Structure, the maximum size for an IP datagram over Ethernet is 1518-bytes. The first Tcpdump capture of the attacks had a snaplen set to 1518-bytes and the results were saved to a file.

The second control test consisted of replaying the captured files and confirming that the IDSs could detect the five replayed attacks. Using a program called Tcpreplay [12], the captured file for each attack was replayed on the network. All IDSs detected the five replayed attacks. The results of the second control test can be found in the highlighted rows labelled 1518-bytes in Tables 3, 4 and 5. The 1518-byte datagram length comprised the upper bound datagram capture length for our testing.

3.3.2. Testing

To generate the necessary network capture test files, the attack tools were executed repeatedly against the target computers. Each time one of the five attack tools was run, the snaplen variable was decreased to reduce the amount of captured datagram payload. Snaplens of 1200, 1000, 759, 500, 200, 64, and 54-bytes were selected for the study. During each attack execution and subsequent network traffic capture, the IDSs were inspected to ensure that the attack was actually detected. All three IDSs detected the attacks as they were captured using varying snaplens. This confirmed that the attacks were executed and that the captured data was derived from actual attack traffic.

Once the data captures were completed for all of the five attack tools, the victim and attacker computers were removed from the network. This avoided contaminating the replayed sessions with spurious responses to the replayed traffic. The captured files were then replayed and the IDS results noted in Tables 3, 4, and 5. For the purposes of our testing, if a specific or generic alarm could be attributed to the intrusion, the replayed session was considered to be successfully detected. For example, Attack C generated a specific alarm on IDS – 1. IDS – 1 identified the intrusion as an IIS printer overflow attack. As the attack was replayed with smaller snaplen sizes, IDS –1 generated a more generic buffer overflow attack alarm (i.e. x86 NOOP). For a snaplen of 64-bytes or less, the attack could not be detected.

A lower bound of 54-bytes was chosen for attack replays. This byte length corresponds to the minimum number of bytes required for a full investigation of a TCP header. This lower bound byte size allows for the replay of all TCP, UDP, and ICMP header information. ICMP normally does not contain any user data. ICMP is used to send messages between computing systems for diagnostic or management purposes. TCP and UDP contain user data in applications such as: e-mail, ftp, and streaming audio.

3.4. Analysis Limitations

Further assumptions should not be extrapolated from these test results. While the results are accurate for the five tools used, they are not intended to be predictive in nature. By modifying a tool, its signature can be changed. As well, many tools offer a variety of configuration options. Our testing used only one configuration. Using different configurations may change the test results.

Our results were based on an Ethernet MTU of 1518-bytes. If a different MTU was used, results would vary. TCP, UDP, and ICMP are the most common protocols seen on the Internet. However, there are many other protocols used on the Internet. Our test results would not necessarily apply to other protocols. The five tools used in this study represent a small sample of the different vulnerability classes. For example, the Mitre Common Vulnerabilities Exposure list contains 2032 entries and 1994 candidates [13].

4 . Results

Legend: √ – Intrusion Detected X – Intrusion Detection Failure – Control Test

IDS – 1					
bytes	**A**	**B**	**C**	**D**	**E**
1518	√	√	√	√	√
1200	√	√	√	√	√
1000	√	√	√	√	√
759	√	√	√	√	√
500	√	√	√	√	√
200	√	√	√	√	√
64	√	√	X	√	√
54	√	√	X	√	√

Table 3: IDS –1 Test Results

IDS – 2					
bytes	**A**	**B**	**C**	**D**	**E**
1518	√	√	√	√	√
1200	X	X	X	√	√
1000	X	√	X	√	√
759	X	√	X	√	√
500	X	√	X	√	X
200	X	X	X	√	X
64	X	X	X	√	X
54	X	X	X	X	X

Table 4: IDS –2 Test Results

IDS – 3					
bytes	**A**	**B**	**C**	**D**	**E**
1518	√	√	√	√	√
1200	√	√	√	√	√
1000	√	√	√	√	√
759	√	√	√	√	√
500	√	√	√	√	√
200	√	√	√	√	√
64	√	√	X	√	√
54	√	√	X	√	√

Table 5: IDS –3 Test Results

Tables 3, 4, and 5 contain the results of our testing. IDS – 1 and IDS – 3 performed similarly during testing. With the exception of Attack C, all attacks were detected regardless of the snaplen used. IDS – 2 did not detect attacks that employed only the TCP protocol (i.e. Attack A, Attack C) at a snaplen of less than 1518 bytes. IDS – 2's failure to detect attacks could be a result of incorrect values in the IP or TCP header fields. The IP header contains a field called "total length" that contains the datagram byte length. The TCP header contains a field called "checksum" that allows corrupted data to be detected. In captures that employed a snaplen of less than 1518-bytes, the datagram length values may not correspond to their actual byte length. Therefore, IDS – 2 may be employing datagram inspection techniques other than simple intrusion signature matching. IDS – 2 may have detected the modified datagrams and due to their incorrect header values either (1) ignored them or (2) determined they would be discarded by the end system and therefore issued no alarm. This hypothesis is supported by the fact that in Attacks A and C (for all replays less than 1518-bytes), IDS – 3 issued IP length mismatch alarms. This did not appear to affect the ability of IDS – 3 to identify the attacks.

At a capture length of 64-bytes, Attack D was detected by all three IDSs. Attack D makes use of the UDP protocol. As shown in Figure 1: Datagram Structure, a UDP header is 42-bytes in length and a capture length of 64-bytes would contain 22-bytes of data. Therefore, the signature for this attack must have been present within this 22-byte data segment.

Attack C exploits a Microsoft IIS printer service buffer overflow. It is an attack that requires a TCP connection to be established to the victim. A capture length of 64-bytes will contain only 10-bytes of data. None of the IDSs were able to detect this attack at a 64 byte or below capture length; therefore, the signature must have been located outside of the 10-byte range.

As the capture length of Attack C was lowered, IDS – 3 issued non-specific alarms. With full datagram capture, Attack C was clearly identified as the IIS printer buffer overflow attack. At lower capture lengths, the identification became more generic as a web attack. Finally, it was interpreted as a NOOP signature. Therefore, the specific portion of the signature must have been located deeper in the datagram.

IDS – 2 detected Attack B at 1000-bytes, but it failed to detect Attack B at 1200-bytes. This would not be expected as the IDS had inspected more bytes of the datagram. This may be a function of the attack tool rather than the IDS. Attack B employs a random attack generator. Therefore, the session that was captured at 1200-bytes must have had signatures in the datagram deeper than the capture length.

4.1. Impact of Partial Datagram Inspection

Our partial IP datagram inspection testing revealed that:

- Capturing only datagram headers requires more than a single byte length. The capture length required for TCP headers is 12-bytes greater than a UDP header.
- Entire classes of attacks can become invisible to IDSs. For example, buffer overflows are a class of attacks that will not be detected by datagram header inspection.
- The fidelity of IDS alarms can be reduced. Even if the attack is detected, the attack specifics might be lost.
- Some IDSs may be sufficiently advanced to ignore corrupt datagrams.

5 . Related Work

Over the last few years, the field of IDS has grown. It is now considered industry best practice to include IDS as a part of a layered network security posture. However, there still has been limited research performed on the topic of IDS testing. Most of the existing work can be categorized as: evasion techniques, breadth of intrusions detected, performance, and robustness.

There are several techniques that employ insertion, evasion, and DoS activities to defeat intrusion detection [14]. Most IDSs are designed to “failopen”. Therefore, IDSs can be adversely affected by DoS attacks. Intentionally inserting data or fragmenting datagrams may cause an IDS to interpret the network traffic differently. IDSs may reject datagrams that would be accepted by end systems. Therefore, IDSs may fail to alarm on attacks that will be successful on end systems. Other methods to defeat IDSs build upon the insertion, evasion, and denial of service techniques. A number of methods can be employed to defeat IDSs such as: subverting active responses, social engineering, operator DoS, state exhaustion, and splitting attacks across multiple systems [15].

Puketza et al, developed a methodology for testing intrusion detection systems [16]. Their research focused on developing a generic methodology to test IDSs based on the following performance objectives: (1) does it detect a broad range of intrusions, (2) how much resources are used, and (3) how resilient is it to stress. Developing specific test cases was challenging. The total number of intrusions is too large to simulate; therefore, they separated intrusions into classes and selected one or more intrusions from each class. This technique is known as equivalence partitioning. They developed a strategy for classifying intrusions that involved categorizing intrusions by: intrusion technique used, systems vulnerability exploited, and their IDS signature. Test cases were then derived from each classification.

DARPA also performed testing on IDSs using a test bed that simulated live network traffic from hundreds of users on thousands of sites [17]. Hundreds of attacks were launched on the network monitored by IDS. A large percentage of the attacks were missed due to the depth at which some protocols were examined and the inability of existing signatures to detect new attacks.

However, IDS testing is somewhat controversial [8]. Some common problems are: the production of artificial attack data, the use of artificial network loads, and creating IDS biased tests. In some cases, the IDS with the strongest detection ability can be put at an unfair disadvantage.

6 . Conclusions

IDS detection capabilities vary even when a small sample of attack tools is used. In some attacks the IDSs only require the header information to detect the intrusion (e.g. scanning). In other attacks, a significant portion of the datagram must be inspected to determine if an intrusion has occurred (e.g. buffer overflow).

Our test results suggest that it is possible to reduce datagram inspection and still detect some attacks. However, depending on the captured byte length: (1) a significant number of attacks may still be missed and (2) a generic alarm may be issued instead of the actual intrusion alarm. A standard capture length cannot be used to inspect protocol headers because various protocols have different header lengths. For example, using the TCP header length of 54-bytes will capture UDP datagram content; however, by using the UDP header length of 42-bytes a portion of the TCP header will be missed.

7 . References

[1] MacLeod, D., Whyte, D. (September 2000). Towards System Survivability using the Single Virtual Enterprise Model and Layered Security through Information Protection Co-ordination Centres. CERT/CC. Available at: http://www.cert.org/research/isw/isw2000/papers/55.pdf

[2] (November 1999). Acquiring and Deploying Intrusion Detection Systems, ITL Bulletin. Available at: http://www.itl.nist.gov/lab/bulletns/nov99.htm

[3] Allen et al. (January 2000). State of the Practice of Intrusion Detection Technologies. Available at: http://www.sei.cmu.edu/publications/documents/99.reports/99tr028/99tr028abstract.html

[4] Stevens, R. (October 1999). TCP/IP Illustrated, Volume 1 The Protocols. Addison-Wesley Professional Computing Series.

[5] (September, 1981). RFC: 791. Internet Protocol Darpa Internet Program Protocol Specification. Available at: ftp://ftp.isi.edu/in-notes/rfc791.txt

[6] (September, 1981). RFC: 793. Transmission Control Protocol Darpa Internet Program Protocol Specification. Available at: ftp://ftp.isi.edu/in-notes/rfc793.txt

[7] Part 3: Carrier sense multiple access with collision detection (CSMA/CD) access method and physical layer specifications. (ISO/IEC 8802-3: 2000 (E)). IEEE Standard for Information technology--Telecommunications and information exchange between systems--Local and metropolitan area networks--Specific requirements. Available at: http://standards.ieee.org/getieee802/802.3.html

[8] Ranum, Marcus. (December, 2001). Experiences Benchmarking Intrusion Detection Systems. Available at: http://www.nfr.com/forum/white-papers/Benchmarking-IDS-NFR.pdf

[9] Network exploit tools available at: http://packetstormsecurity.nl

[10] NetCat is available at: http://www.l0pht.com/

[11] Tcpdump is available at: http://www.tcpdump.org

[12] Tcpreplay is available at: http://packetstormsecurity.nl

[13] http://cve.mitre.org

[14] Ptacek, T., Newsham, T. (January 1998) Insertion, Evasion and Denial of Service: Eluding Network Intrusion Detection. Available at: http://citeseer.nj.nec.com/ptacek98insertion.html

[15] Cohn, F. (December, 1997). 50 Ways to Defeat Your Intrusion Detection System. Available at: http://all.net/journal/netsec/9712.html

[16] Puketza et al. (September 27, 1996). A Methodology for Testing Intrusion Detection Systems. Available at: http://citeseer.nj.nec.com/puketza96methodology.html

[17] Lippman et al. (1999). The 1999 DARPA Off-Line Intrusion Detection Evaluation. Available at: http://www.ll.mit.edu/IST/ideval/pubs/2000/1999Eval-ComputerNetworks2000.pdf

8 . Acronyms

DoS	Denial of Service
ICMP	Internet Control Message Protocol
IDS	Intrusion Detection System
IP	Internet Protocol
IT	Information Technology
MAC	Media Access Controller
MTU	Maximum Transfer Unit
PC	Personal Computer
TCP	Transmission Control Protocol
UDP	User Datagram Protocol

This page has been deliberately left blank

Page intentionnellement blanche

Click here to view PowerPoint presentation; Press Esc to exit

Application of Genetic Optimization and Statistical Analysis for Detecting Attacks on a Computer Network

Victor A. Skormin, Douglas H. Summerville, James S. Moronski
Watson School of Engineering, Electrical and Computer Engineering
Binghamton University, Binghamton, NY 13902, USA

James L. Sidoran
Air Force Research Laboratory/IFGB
525 Brooks Road, Rome, NY 13441-4505, USA

Abstract

Modern computer networks are vulnerable to information attacks. Detection of a distributed attack at the earliest possible stage of its development presents an equally important and difficult task for a network administrator. This paper presents an approach, utilizing statistical analysis of the data traffic through the network that would enhance the administrator's ability to solve this task. The developed data mining technology is based on a modified cluster analysis aided by genetic optimization. The iterative Bayesian technique is employed for the real-time calculation of the probability of an attack subject to the current pattern of data traffic through the network. The approach is implemented in software and demonstrated on real data obtained on an experimental computer network.

1. Introduction

The vulnerability of modern computer networks to information attacks is of great concern to any organization utilizing computer networks. With the increase of size, interconnectivity and number of users, networks are becoming increasingly vulnerable to newly developed direct and remote threats and attacks compromising the integrity, confidentiality or availability of a network, and consequently information.

Denial of Service (DoS) attacks making crucial network resource and/or information unavailable, are among the most common attack types. DoS attacks are both effective and easy to deploy. DoS attacks threaten all systems connected to the Internet, including servers, clients, routers, and firewalls. In a DoS attack scenario, the attacker sends malicious traffic to the target with the aim of crashing, crippling, or jamming communication between the target system and legitimate users.

Fundamentally, there are three DoS attack scenarios, all of which effectively disable the target and prevent its legitimate use. In the first scenario, the attacker exploits bugs in network software implementations, crashing or disabling the target's communication processing capabilities. The second attack type is aimed at weaknesses in network protocols, aiming to overload the system's communication resources, which disconnects the target from the outside world. The third type of attack exploits the limited network bandwidth to the target, inundating it with enormous volumes of traffic.

Clearly, all three forms of attack have the common result of preventing legitimate use of the target system. Although patchwork solutions have been developed for many of the DoS attacks currently identified, new attacks are continually being developed. The purpose of connecting computer systems to a network is to provide access for their legitimate use. Although network protocols can be made more secure, any mechanism that allows outside access to a system can be exploited and makes that system vulnerable to attack. Thus, protecting networked systems requires accepting the dynamic, uncontrollable, and potentially hostile environment in which they exist and developing protection mechanisms that can cope with this environment.

Paper presented at the RTO IST Symposium on "Real Time Intrusion Detection", held in Estoril, Portugal from 27-28 May 2002, and published in RTO-MP-101.

Current computer security systems provide some degree of protection of information attacks and enhance the decision-making ability of the network administrator. These functions are performed by a number of independent system components requiring an enormous amount of distributed and specialized knowledge, and consequently, computations. As a rule, these systems represent a bottleneck with regard to throughput, speed, reliability and flexibility of a network

Unlike hardware failures of network components, information attacks unravel in time, propagating through the network and affecting the increasing number of users and hosts. A network operator could detect these undesirable developments at relatively early stages, and undertake important decisions alleviating the attack. These decisions typically reflect experience and intuition of human operator and consequently are based on his/her subjective interpretation of the current status of the network. The dynamic nature of information attacks and ability to monitor status of a network facilitate the development of mathematically justified, more efficient approaches to the detection of attacks providing important insight to human operators. This paper is aimed at the application of advanced statistical analysis for the objective assessment of the network status and detection of unique changes in the status indicative of information attacks. Implementation of this approach can result in an information security tool facilitating early detection of the DoS-type attacks and providing on-line support of operators' decisions.

2. Quantified Representation of a Network Status

A modern computer network constitutes a complex dynamic system containing many relatively independent, highly interconnected, dynamic components. While the structural complexity of such a network is not to be questioned, its dynamic nature is not that obvious. Indeed, electronics-based network hardware has negligible dynamics. However, every data processing, digital communication, and data interpretation procedure, implemented in software, has finite execution time. Within the network, operation of every logical unit can be characterized by the data traffic flow through this unit that comprises data packets of various sizes. The composition of data flow could be quantified by percentages of large, medium and small size packets. The packets may be accumulated in queues and are processed in some order that results in the effect of "inertia" responsible for the network dynamics. Unlike classical view of system dynamics, dynamics of a computer network is both system- and input-dependent. System dependence of network dynamics is related to the throughput of particular modules of the network. Input dependence manifests itself by time-varying number of packets to be processed and their composition, and specific computational tasks.

It is understood that while the network configuration and throughput of its components are not affected by information attacks, they do not carry any useful information for our purpose. However, information attacks cause gradual changes in the volume and composition of particular data flows within the network, ultimately leading to a complete denial of service. While periodic changes of the flows volumes and compositions could be observed under normal operation of the network on the 24-hour and 7-day scales, it is expected that changes in the flow volumes and compositions caused by information attacks follow very different statistical patterns. This creates the opportunity for differentiating "normal" fluctuations from those caused by attacks. These realities have led the authors to the following conclusion: *dynamic properties of a computer network, representing its immediate status, could be best described in a specially defined state space that reflects not the configuration of the network hardware or software, but the volumes and compositions of data flows through particular modules of the network, and their rates of change.*

Assume that a computer network consists of M interrelated processing modules, and the m-th module has the data flow characterized by

- volume, i.e. number of bytes per second, - z_m,
- composition of the data flow, i.e. the percentage of large, medium and small packets – p_m, r_m, s_m, such that $p_m+r_m+s_m=1$, and
- time derivatives (rates of change) of these variables – z'_m, p'_m, r'_m, s'_m, where $m=1,2,\ldots,M$.

For convenience, all "state variables" of the computer network could be assembled under the state vector

$$X = [z_m, p_m, r_m, s_m, z'_m, p'_m, r'_m, s'_m, \ m=1,2,\ldots,M] \qquad (2.1)$$

It could be seen that the attempt to quantify the status of a computer network leads to the definition of the state space of high dimension. It is also understood that *X* is a random vector and its realization *X(k)* represents the an immediate status of the network at the discrete time moment *k*=1,2,3…. The availability of a network monitoring system allows us to accumulate a database, {*X(k)* , *k*=1,2,3…, *N*}, representing the network status (operation) over some period of time. Although the established state space and database may not be sufficient for the formulation of the network control problems, it can provide sufficient information for the detection of specific changes in the network status caused by information attacks. Fortunately, attack detection/recognition is a much simpler task than control that may concentrate on the analysis of some "bottlenecks" of the network. Application of statistical analyses facilitates the detection of the "informative" components of the state vector thus resulting in the significant simplification of the attack detection/prediction problem, and consequently, definition of simplistic recognition rules. It could be achieved if, and only if, the available database contains both the normal operational data, and the information accumulated during attack of the known type, i.e. has the format {*X(k)*, *Q(k)*, *k*=1,2,3…, *N*}, where *Q(k)* is a flag indicating the presence of the attack at the discrete time moment *k*.

Recall that the particular realizations of the database are random vectors. Application of a statistical approach to the detection/prediction problem results in "averaging out" the effects of such unimportant factors and fluctuations of variables *X* during normal operation of the network, and detecting abnormal trends in the network status data, that could be interpreted as an incipient information attack. The following mathematical techniques are consistent with the statistical nature of the attack detection/prediction problem.

3. Cluster Analysis and Genetic Optimization

Cluster analysis is a group of statistical techniques facilitating the detection of informative components of what could be a very extensive database. It is clear that this task cannot be accomplished without relevance to some decision-making or a classification problem. We will visualize the database as a combination of realizations of real status variables, X , and a binary class indicator, Q:

$$\{X(k), Q(k)\} = \{x_1(k), x_2(k), x_3(k), \ldots, x_n(k), Q(k)\} \qquad (3.1)$$

where $k = 1,2,3,\ldots,N$ is the realization index, and $Q(k)$ can have only two alternative values, "**a**" or "**b**". Then the classification rule is established on the basis of some function defined in the X space, $F[X]$, such that, generally, $F[X] \le 0$ for the majority of realizations when $Q(k) = "\text{a}"$ and $F[X] \succ 0$ for the majority of realizations when $Q(k) = "\text{b}"$, or in terms of conditional probabilities,

$$P\{F[k] \le 0 \ / \ Q[k] = "\text{a}"\} \ \succ \ P\{F[k] \le 0 \ / \ Q[k] = "\text{b}"\} \qquad (3.2)$$

where $P\{A / B\}$ denotes the probability of event *A* subject to the occurrence of event *B*.
It is also understood that the classification problem does not have a unique solution, and there is a wide class of functions $F[X]$ that could satisfy condition (3.2) to a greater or a lesser extent. A simplification of the classification rule requires reducing the number of the components of vector X to the necessary minimum by choosing the smallest group of informative components that, in combination, allow for achieving reliable classification.

Selection of the informative components implies that contributions of particular groups of components of vector X to the classification are to be evaluated, and the most contributive group(s) be chosen for the definition of the classification rule. One can realize that in order to achieve the required discrimination power of the selection procedure, the groups must be small, and in order to consider combined effects of several variables must include at least two variables [1, 2]. Consider all possible combinations of two variables taken out of *n*, where *n* is the dimension of vector X . It could be said now that the classification problem, originally defined in the space X , now will be considered on particular two-dimensional subspaces, $x_i \quad x_j$, where $i, j = 1,2,\ldots,n,$ and $i \ne j$.

Assume that the entire array of points, marked as "**a**" or "**b**", defined in the *n*-dimensional space X by the database (3.1), is projected on particular two-dimensional subspaces (planes). Let us visualize possible distributions of these projections. Figure 1 below illustrates a subspace that has no potential for the development of a classification rule due to the fact that points marked as "**a**" and "**b**" are distributed quite uniformly in this plane. The subspace of Figure 2 indicates a certain degree of separation between points "**a**" and "**b**" and, therefore, should be viewed as informative. Figures 4, 5, 6 also illustrate possible cases of separation pattern in informative subspaces.

As shown in Figures 4, 5, 6, a correlation ellipse, properly defined in the particular informative subspace, presents an ideal choice of the separating function. Figure 3 indicates that size, shape, position, and orientation of such an ellipse are defined by five parameters: coordinates of two focal points, $[\alpha_1, \beta_1]$, $[\alpha_2, \beta_2]$ and the constant δ, such that for any points of the ellipse, $[x_i, x_j]$, the following equation holds,

$$\sqrt{(x_i - \alpha_1)^2 + (x_j - \beta_1)^2} + \sqrt{(x_i - \alpha_2)^2 + (x_j - \beta_2)^2} = \delta \qquad (3.3)$$

Similarly, equations

$$\sqrt{(x_i - \alpha_1)^2 + (x_j - \beta_1)^2} + \sqrt{(x_i - \alpha_2)^2 + (x_j - \beta_2)^2} \le \delta \qquad (3.3a)$$

$$\sqrt{(x_i - \alpha_1)^2 + (x_j - \beta_1)^2} + \sqrt{(x_i - \alpha_2)^2 + (x_j - \beta_2)^2} \succ \delta \qquad (3.3b)$$

represent any point $[x_i, x_j]$ within and outside the ellipse.

Consider the problem of the optimal definition of parameters $[\alpha_1, \beta_1, \alpha_2, \beta_2, \delta]$ of a correlation ellipse for a particular separation pattern in the plane comprising variables x_i and x_j. According to condition (3.2), this problem could be interpreted as the minimization of a loss function that includes a "penalty" for any point "a" outside the ellipse, $R^a(k) = R^a[x_i^a(k), x_j^a(k)]$, and a "penalty" for any point "b" within the ellipse, $R^b(k) = R^b[x_i^b(k), x_j^b(k)]$, i.e.

$$L(\alpha_1, \beta_1, \alpha_2, \beta_2, \delta) = \sum_{k=1}^{N^a} R^a(k) + \sum_{k=1}^{N^b} R^b(k) \qquad (3.4)$$

where N^a and N^b are number of points "a" and "b" in the database. Intuitively, these penalties are defined as follows:

$$R^a(k) = \begin{cases} 0, \text{ if point } [x_i^a(k), x_j^a(k)] \text{ satisfies condition (3.3a)} \\ \dfrac{1}{[x_i^a(k) - \mu_i^a]^2 + [x_j^a(k) - \mu_j^a]^2}, \text{ if point } [x_i^a(k), x_j^a(k)] \text{ satisfies condition (3.3b)} \end{cases}$$

and

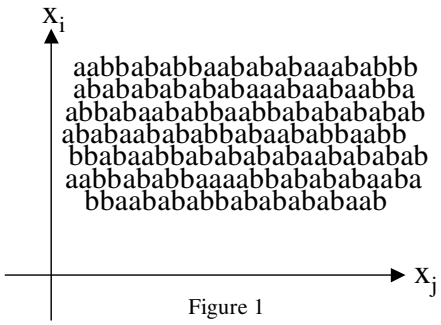


Figure 1

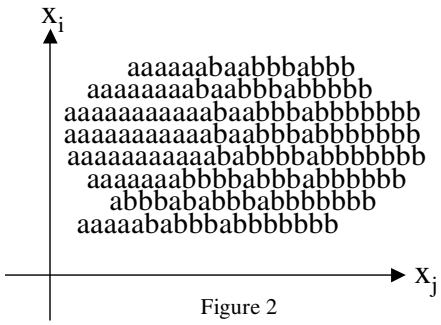


Figure 2

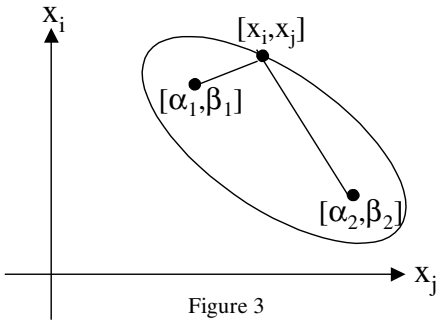


Figure 3

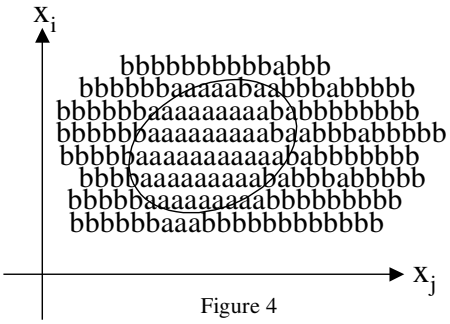


Figure 4

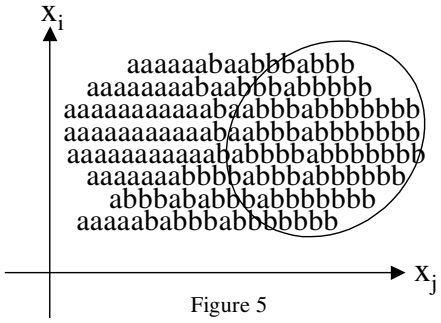


Figure 5

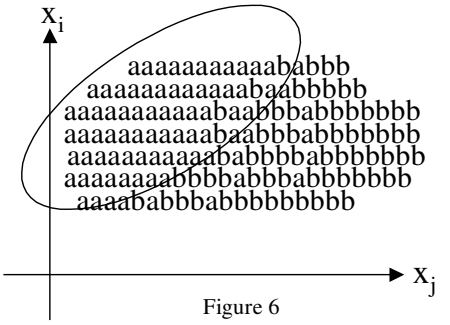


Figure 6

$$R^b(k) = \begin{cases} 0, \text{ if point } \left[x_i{}^b(k), x_j{}^b(k)\right] \text{ satisfies condition (3.3b)} \\ \dfrac{1}{\left[x_i{}^b(k) - \mu_i{}^b\right]^2 + \left[x_j{}^b(k) - \mu_j{}^b\right]^2}, \text{ if point } \left[x_i{}^b(k), x_j{}^b(k)\right] \text{ satisfies condition (3.3a)} \end{cases}$$

where $\left[\mu_i{}^a, \mu_j{}^a\right]$ and $\left[\mu_i{}^b, \mu_j{}^b\right]$ are coordinates of the geometric centers of points "**a**" and points "**b**" distributed in the plain $x_i \quad x_j$. Such a choice of penalty functions places highest emphasis on the points in the immediate vicinity of geometric centers.

It could be seen that the loss function (3.4) is not only nonlinear but also discontinuous with respect to the unknown parameters of the separation ellipse $\left[\alpha_1, \beta_1, \alpha_2, \beta_2, \delta\right]$. Therefore our attempt to obtain the numerical values of these parameters by minimizing this loss function leads to a highly nonlinear multivariable optimization problem that does not have an analytical solution. Moreover, finding its global solution numerically would also be a very difficult task. Such an optimization problem presents an ideal application for a genetic optimization procedure that combines the advantages of both direct and random search [3, 4]. Application of a genetic algorithm results in the definition of an ellipse that indeed contains the largest possible number of points "**a**", N^{aa}, and the smallest possible number of points "**b**", N^{ab}. Then the "goodness" of the ellipse-based separating rule could be characterized by the following two quantities:

$$P_{in}\{a/a\} \approx \frac{N^{aa}}{N^a} \text{ and } P_{in}\{a/b\} \approx \frac{N^{ab}}{N^b} \tag{3.5}$$

representing the probabilities of a point "**a**" and a point "**b**" to be found *within* the ellipse, see Figure 4.

Should we assume that the obtained classification rule, reflecting some compromise solution, could not be further improved? In our experience an alternative classification rulecould be obtained by establishing an ellipse containing as many points "**b**", N^{bb}, and as few points "**a**", N^{ba}, as possible. This task is accomplished by the appropriate modification of the penalty functions. The resultant separating rule is characterized by:

$$P_{out}\{a/a\} \approx 1 - \frac{N^{ba}}{N^a} \text{ and } P_{out}\{a/b\} \approx 1 - \frac{N^{bb}}{N^b} \tag{3.6}$$

representing the probabilities of a point "**a**" and a point "**b**" to be found *outside* the ellipse, see Figure 5. The final selection of a separating rule should implies the comparison of ratios

$$\rho_{in} = \frac{P_{in}\{a/a\}}{P_{in}\{a/b\}} \text{ and } \rho_{out} = \frac{P_{out}\{a/a\}}{P_{out}\{a/b\}} \tag{3.7}$$

obtained for the "inside the ellipse" and "outside the ellipse" rules, and choosing the rule that results in the largest ρ value.

While the application of the genetic optimization results in the optimal characteristics of both ellipses, the final choice of ratio (3.7) provides an explicit informativity measure of the subspace $x_i \quad x_j$ where the ellipses are established thus facilitating the selection of the most informative subspaces. Unfortunately, genetic algorithms emulating evolutional developments in nature are too slow for being applied to the entire multitude of possible combination of two components out of *n*, especially if *n*>20. Therefore, this task should be performed on the basis of a less rigorous informativity measure, the weighted average distance between points "**a**" and "**b**" in the particular subspaces,

$$\rho_{ij} = \frac{1}{N^a N^b \sigma_i \sigma_j} \sum_{k=1}^{N^a} \sum_{m=1}^{N^b} \sqrt{\left[x_i{}^a(k) - x_i{}^b(m)\right]^2 + \left[x_j{}^a(k) - x_j{}^b(m)\right]^2} \tag{3.8}$$

where

σ_i are σ_j are standard deviations of variables x_i and x_j.

Finally, the proposed procedure implies

1. Definition of all two-dimensional subspaces of the space *X*
2. Computation of the informativity measure (3.8) for every subspace $x_i \quad x_j$
3. Selection of a number *M* of the most informative subspaces
4. Selecting one of the *M* informative subspaces
5. Establishing the "inside the ellipse" classification rule by the application of a genetic optimization procedure, and computation of the ρ_{in} value for this subspace
6. Establishing the "outside the ellipse" classification rule by the application of a genetic optimization procedure, and computation of the ρ_{out} value for this subspace
7. Selection of the classification rule that has the largest ρ value for this subspace, and return to Step 4, until the list of informative subspaces will be exhausted.

Proliferation of genetic optimization algorithms, possessing the advantages of known random and direct search optimization procedures, combined with the availability of high performance computers alleviates major obstacles in the way of the solution of multivariable nonlinear optimization problems. The following figure illustrates application of a genetic algorithm to the solution of a classification problem mentioned above.

The algorithm proceeds as follows. Combinations of an ellipse parameters represented by vector $P=\left[\alpha_1,\beta_1,\alpha_2,\beta_2,\delta\right]$ form 5-dimensional space *S*. Since those parameters have bounded values, consistent with the initial variables X of the classification problem, an acceptable solution will be within a subspace, whose boundaries are defined by the following inequality $P_1 \le P \le P_2$. The algorithm forms an initial grid within this subspace by generating K uniformly distributed points P_i, $(i=1,2,\ldots,K)$, that represent possible solutions of the optimization problem.

Each point P_i of the initial grid represents an ellipse that is being assessed for the classification rule by computing the corresponding loss function $L(P_i)$ defined in (3.4), where

$$P_i = (\alpha_{1i}, \beta_{1i}, \alpha_{2i}, \beta_{2i}, \delta_i).$$

After the loss function $L(P_i)$ has been calculated at each point of the initial grid, the first generation of ellipses is formed by selecting $N_g < K$ points P_j, $(j=1,2,\ldots,N_g)$ with the smallest values of $L(P)$.

The process of parenting involves producing N_o offsprings per each of the $(N_g-1)*N_g/2$ possible pairs of points from the initial generation. Each offspring is a new point in the 5-dimensional space located on the line connecting its parents P_i and P_j $(i \neq j)$, and selected in a random fashion. At the end of this process the total population becomes $N_o*(N_g-1)*N_g/2 + N_g$ points.

Parenting is followed by the mutation stage. Each point from the expanded population produces N_m mutations (points generated randomly in the immediate area). Thus, the total population count grows to

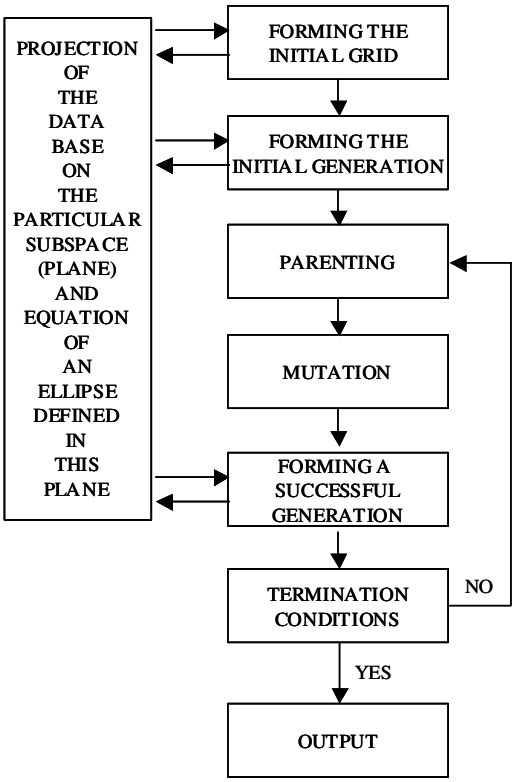


Figure 7. Genetic optimization for optimal ellipse definition

$(N_m+1)(N_o*(N_g-1)*N_g/2+ N_g)$ points. A successive generation is formed by computing the loss function for all the points and selecting N_g points with the smallest values of the loss function $L(P)$.

The processes of parenting, mutation and forming a successive generation are repeated until a termination condition is satisfied. The optimization routine produces the output in the form of a vector

$$P^{OPT}=(\alpha_1^{OPT}, \beta_1^{OPT}, \alpha_2^{OPT}, \beta_2^{OPT}, \delta^{OPT})$$

which represents an ellipse resulting in the best separation rule attainable for the particular combination of two variables of the initial classification problem.

4. Bayesian Decision Making

Statistical analysis of the network data, representing normal operation of the network and its operation under known DoS attacks, results in the extraction and formalized representation of knowledge of various effects of the attack on the network. This information is attack- , host- and network-specific, and can be used for early attack detection, and potentially, for the type of attack recognition. The following mathematical framework is suggested for the attack detection scheme.

Assume that the preliminary cluster analysis utilizing the informativity criterion (3.8) has resulted in the set of M two-dimensional informative subspaces. Then the set of M respective, either "inside the ellipse" or "outside the ellipse" classification rules, $R_i[X(k)]$, i =1,2,3,..M, was developed by the application of genetic optimization. One

can realize that each rule utilizes only those two components of vector X that constitute the i-th informative subspace. For simplicity, assume that each classification rule is designed to return a negative value for the majority of points $X(k)$ representing an attack situation (and marked by an "a"). It is expected that every vector X^a, representing an attack, and every vector X^b, representing a normal network operation, would satisfy only some classification rules but not all of them. Consider the following random events:

$$
\begin{aligned}
&E_1:\ R_1[X(k)]\le 0 \cap R_2[X(k)]\le 0 \cap R_3[X(k)]\le 0 \cap R_4[X(k)]\le 0 \cap \dots R_M[X(k)]\le 0\\
&E_2:\ R_1[X(k)]> 0 \cap R_2[X(k)]\le 0 \cap R_3[X(k)]< 0 \cap R_4[X(k)]\le 0 \cap \dots R_M[X(k)]\le 0\\
&E_3:\ R_1[X(k)]\le 0 \cap R_2[X(k)]> 0 \cap R_3[X(k)]\le 0 \cap R_4[X(k)]\le 0 \cap \dots R_M[X(k)]\le 0\\
&E_4:\ R_1[X(k)]> 0 \cap R_2[X(k)]> 0 \cap R_3[X(k)]\le 0 \cap R_4[X(k)]\le 0 \cap \dots R_M[X(k)]\le 0\\
&E_5:\ R_1[X(k)]\le 0 \cap R_2[X(k)]\le 0 \cap R_3[X(k)]> 0 \cap R_4[X(k)]\le 0 \cap \dots R_M[X(k)]\le 0\\
&\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\\
&E_L:\ R_1[X(k)]> 0 \cap R_2[X(k)]> 0 \cap R_3[X(k)]> 0 \cap R_4[X(k)]> 0 \cap \dots R_M[X(k)]> 0
\end{aligned}
\qquad (4.1)
$$

representing specific combinations of classification rules satisfied by every vector $X(k)$. First, note that $L=2^M$. Probabilities of these events, evaluated separately for vectors X^a and X^b, constitute the following set of conditional probabilities instrumental for the decision making procedure:

$$P\{E_i / a\} \text{ and } P\{E_i / b\},\ i = 1,2,\dots, L \qquad (4.2)$$

These off-line tasks could be viewed as the development of a cluster model of the network and constitute the "learning" process.

Now consider the utilization of the cluster model for on-line attack detection. Note that at this point, we will consider any abnormal state of the network as a manifestation of the incipient attack. Assume that the probability of attack on the network has some initial value, established according to the existing statistics, $\gamma[0]$, and therefore the probability of normal operation, $\lambda[0] = 1 - \gamma[0]$.

Assume that vector $X(k) = \left[x_1(k), x_2(k), x_3(k), \dots, x_n(k)\right]$ represents the most recent state of the network. Numerical values of components of this vector, applied to the classification rules $R_i[X]$, i=1,2,..,M, results in a particular combination of numerical values

$$R_1[X(k)],\ R_2[X(k)],\ R_3[X(k)],\ \dots,\ R_M[X(k)]$$

that could be identified as the occurrence of one of the events (4.1), for example, E_j. Now the availability of conditional probabilities (4.2) facilitates the application of Bayesian approach for the re-evaluation of the probability of the attack (i.e. the probability of the point $X(k)$ to be marked by an "**a**") subject to the occurrence of the event E_j, $P\{a / E_j\}$. One can realize that unconditional probabilities, $\mathrm{P}\{\mathrm{a}\} + \mathrm{P}\{\mathrm{b}\} = 1$, therefore

$$P\{a / E_j\} P\{E_j\} = P\{E_j / a\} P\{a\} \text{ and } P\{E_j\} = P\{E_j / a\} P\{a\} + P\{E_j / b\} P\{b\},$$

and the required probability can be expressed as,

$$P\{a / E_j\} = \frac{P\{E_j / a\} P\{a\}}{P\{E_j / a\} P\{a\} + P\{E_j / b\} P\{b\}} = \frac{\gamma[0] \cdot P\{E_j / a\}}{\gamma[0] \cdot P\{E_j / a\} + \lambda[0] \cdot P\{E_j / b\}} \qquad (4.3)$$

Computation (4.3) results in an updated value of the probability of attack, $P\{a / E_j\}$ that could be compared against some arbitrary defined threshold value. A warning message indicating an incipient attack should be issued if the probability of attack exceeds the threshold. This completes one cycle of the proposed procedure. At the next cycle, the "prior" probabilities of attack and the normal operation are defined as,

$$\gamma[k] = P\{a / E_j\} \text{ and } \lambda[k] = 1 - P\{a / E_j\}$$

and the new value of the network state vector,

$$X(k+1) = [x_1(k+1), x_2(k+1), x_3(k+1), \ldots, x_n(k+1)]$$

is to be analyzed with the consequent computation of probabilities $\gamma[k+1]$ and $\lambda[k+1]$. This procedure, intended for continuous real-time application, is capable of providing a timely and objective information to the network operator providing mathematically justified support of his/her decisions.

5. Definition of the "Normal" Status of a Computer Network

The definition of the "normal" status of a computer network depends on the particular configuration and patterns of usage of the specific network to be protected. For a given network, the availability of a network monitoring system allows us to accumulate a database, representing the network status observed during periods of both "normal" operation as well as specific attack scenarios. This database should provide sufficient information for the detection of specific changes in the network status caused by information attacks. Recall that each database record contains data representing the network status at a discrete time *k* and has the format {*X(k)*, *Q(k)*}, where *X(k)* is the state vector representing the status of the network at the discrete time *k* and *Q(k)* is a flag indicating the presence or absence of an attack at time *k*.

The dynamic properties of a computer network that represent its status are described by the properties of the network traffic flows through its particular modules. These properties include the volumes, compositions and rates of change of the traffic flows. Although these parameters fluctuate during normal periods of network operation, it is possible to distinguish a particular information attack based upon the unique effect it has on these parameters.

The experimental network chosen for this study is representative of many intranet configurations currently in use. The specific configuration of the experimental network is shown in Figure 8. It consists of a small local area network (LAN) with a connection to the Internet provided through a router, which lies outside of the administrative domain of the experimental setup. The connection to the router thus represents the "gateway connection" through which all external traffic enters the experimental network. This figure can be generalized to represent nearly any computer network. We have placed the network monitoring station at the gateway connection of the network to be able to easily monitor all traffic flows into the network.

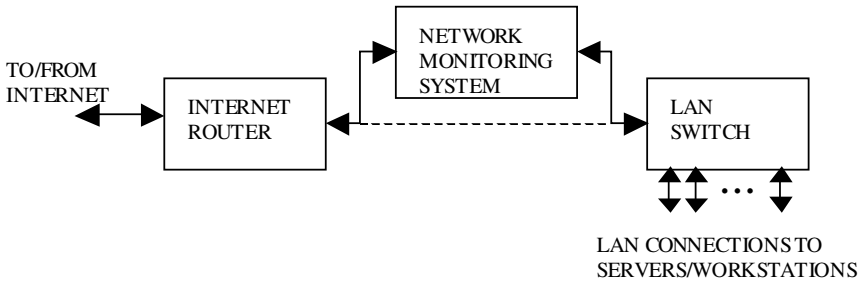


Figure 8. Experimental network configuration

Our experimental setup consists of a 100 Mbps switched Ethernet network interconnecting several SUN UltraSparc computers, some of which are client workstations while others serve as Web and file servers. The network monitoring systems consists of a SUN UltraSparc workstation with two network interface cards placed inline at the network gateway connection. This system monitors network packets both entering and leaving the network. Since the experimental setup uses TCP/IP protocols, the state space of the network has been defined in terms specific to these protocols. State variables represent measured quantities of the aggregate network traffic, using a variety of decompositions. For each measured quantity, its time derivative is also used as a state variable.

Information attacks usually attack a specific protocol. For example, TCP SYN attacks target the TCP protocol, while the Ping Flood attack targets IP control protocol. The aggregate network traffic is decomposed into streams identified by network protocol. The fractions of total packets utilizing the TCP and UDP protocols are measured.

The aggregate network traffic is then characterized by its general flow characteristics. The total volume of network traffic is measured, in bytes per second (normalized to the maximum rate allowed by the network). We also measure traffic rate in packets per second, (normalized to the maximum packet rate for the network). The average packet length may provide an indication of irregular packet flows that could be a sign that an attack is in progress. The traffic stream is decomposed into percentages of large, medium, and small packets. Without loss of generality, we have considered 100 bytes or less to be a small packet, and 500 bytes or greater to be a large packet; all others are considered medium packets.

Data Traffic Variables Subjected to Statistical Analysis

Variable	Description	Comment
x_1	fraction of packets utilizing TCP protocol	relative units
x_2	fraction of packets utilizing TCP protocol, rate of change	relative units
x_3	fraction of packets utilizing UDP protocol	relative units
x_4	fraction of packets utilizing UDP protocol, rate of change	relative units
x_5	fraction of large (>500 bytes) packets in the flow	relative units
x_6	fraction of large (>500 bytes) packets in the flow, rate of change	relative units
x_7	fraction of medium packets in the flow	relative units
x_8	fraction of medium packets in the flow, rate of change	relative units
x_9	fraction of small (<100 bytes) packets in the flow	relative units
x_{10}	fraction of small (<100 bytes) packets in the flow, rate of change	relative units
x_{11}	total volume of traffic, normalized	packets per sec
x_{12}	total volume of traffic, normalized, rate of change	relative units
x_{13}	fraction of TCP packets that are control type packets	relative units
x_{14}	fraction of TCP packets that are control packets, rate of change	relative units
x_{15}	fraction of IP packets that are control type packets	relative units
x_{16}	fraction of IP packets that are control type packets, rate of change	relative units
x_{17}	fraction of total packets that are control type packets	relative units
x_{18}	fraction of total packets that are control packets, rate of change	relative units
x_{19}	traffic volume, normalized	bytes per second
x_{20}	traffic volume, , rate of change	relative units
x_{21}	fraction of IP packets that are fragments	relative units
x_{22}	fraction of IP packets that are fragments	relative units
Q = ”A”	indication of a known attack	
Q = “B”	normal operation of the network	

When network messages are too large for network packets, the message is fragmented and transmitted into several packets. Some attacks, such as Ping Flood, may be characterized by a large number of fragmented messages. The ratio of fragmented traffic to total traffic is measured, as well as the rate of change of the ratio.

Most network protocols identify both data packets, which carry user application data, and control type packets, which are used to implement the protocol itself. Many attack scenarios exploit holes in the protocol architecture or bugs in a specific implementation of a protocol. Thus, the percentage of packets that are control type packets are measured for each protocol. In the TCP protocol, for example, we classify a packet as a control packet if the SYN, FIN, or RESET flags are set. We also measure the fraction of total traffic that represents control type packets.

The above table summarizes designation of particular variables – components of the database.

6. Experimental Results and their Interpretation

To evaluate the proposed approach to attack detection in computer networks, we have chosen the two most common types of denial of service attacks, TCP SYN attack and Ping Flood attack, that were deployed in an experimental computer network. A software implementation of each attack was written in C++ and launched from computers outside of the experimental network. Attack packets were tagged (utilizing the unused type of service field in the IP packet header) for classification by the network monitoring station.

For each attack type, a database of network status vectors was collected over a 24 hour period at 1 second intervals. During this 24 hour period, two attacks were launched at arbitrary times. Thus, the database for each attack contained vectors representing both the normal status of the network and the status of the network during an attack.

TCP SYN Attack

The results of cluster analysis of the "attack/normal" data traffic are shown in Figs. 9.1-9.3. The informative components of the network status chosen by the cluster analysis procedure are fraction of small packets (x_9), fraction of control packets (x_{17}), fraction of TCP packets (x_1), and fraction of UDP packets (x_3). These were combined into the informative subspaces (x_9,x_{17}), (x_3,x_{17}), and (x_1,x_{17}). TCP SYN attacks are characterized by a large number of TCP control packets (SYN packets) flooding the target network. These packets are small (generally 40 bytes). Each figure shows the ellipse chosen by the cluster analysis procedure plotted against the data. In all cases, a point inside the ellipse is classified as an attack point.

From Figure 9.1, it is clear that during a TCP SYN attack there is both a high percentage of small packets and a high percentage of control packets, consistent with the nature of the attack. Since nearly all control packets are small, the fraction of control packets rarely exceeds the fraction of small packets, explaining the near absence of points above the line $x_{17}=x_9$. Attack points represent a high fraction of both control and small packets; as these fractions approach the maximum they are nearly equal and appear closely spaced around the line $x_{17}=x_9$. There are a few non-attack points that exhibit both a high fraction of control packets and a high-fraction of small packets. However, unlike a TCP SYN attack scenario, these points represent periods when the network was relatively quiet and thus generally will not be classified as attack points in the other subspaces. This illustrates the increased classification accuracy gained by considering several informative subspaces.

Figures 9.2 and 9.3 show the second and third most informative subspaces selected by the cluster analysis, respectively. Since TCP SYN attack is characterized by a high percentage of TCP control packets, it is counterintuitive that the subspace (x_3,x_{17}), representing the fraction of control packets vs. the fraction of UDP packets, is considered "more informative" than subspace (x_1,x_{17}), representing the fraction of control packets vs. the fraction of TCP packets. One possible explanation is given as follows. The subspace in Figure 9.3 indicates that during an attack there is both a high percentage of TCP packets and a high percentage of control packets in the network. This is a situation that can also occur normally, however, especially when the network is experiencing a very light load. Figure 9.2 illustrates that during an attack, the flood of TCP SYN packets causes a decrease in the flow of UDP packets due to the high rate of control packets consuming network bandwidth. This combination, however, may not occur as frequently during normal operation, even when the network is experiencing a light load. Thus, cluster analysis has the ability to infer subtle relationships between network status variables which may not be obvious and which may indeed be counterintuitive.

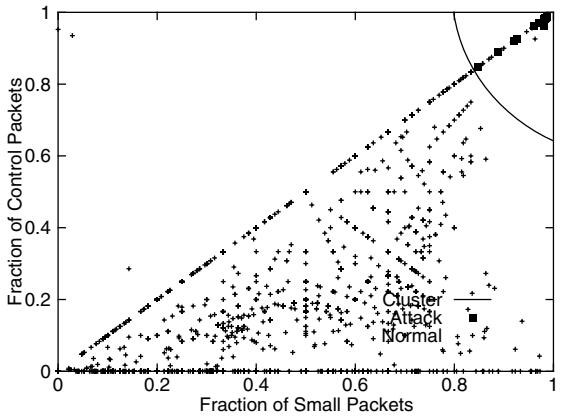


Figure 9.1

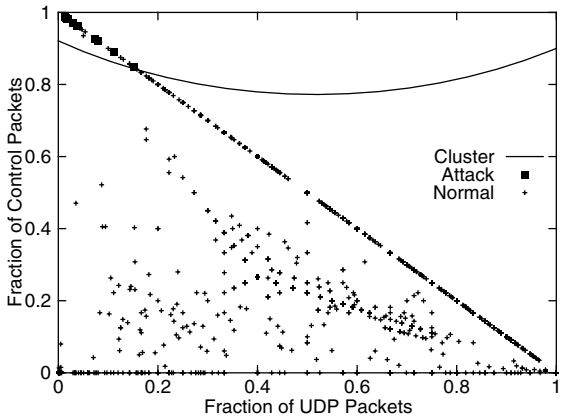


Figure 9.1

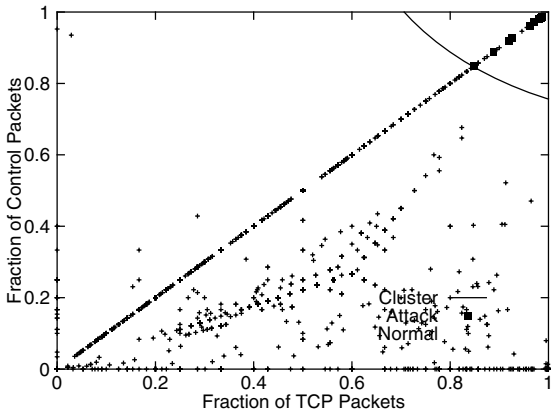


Figure 9.2

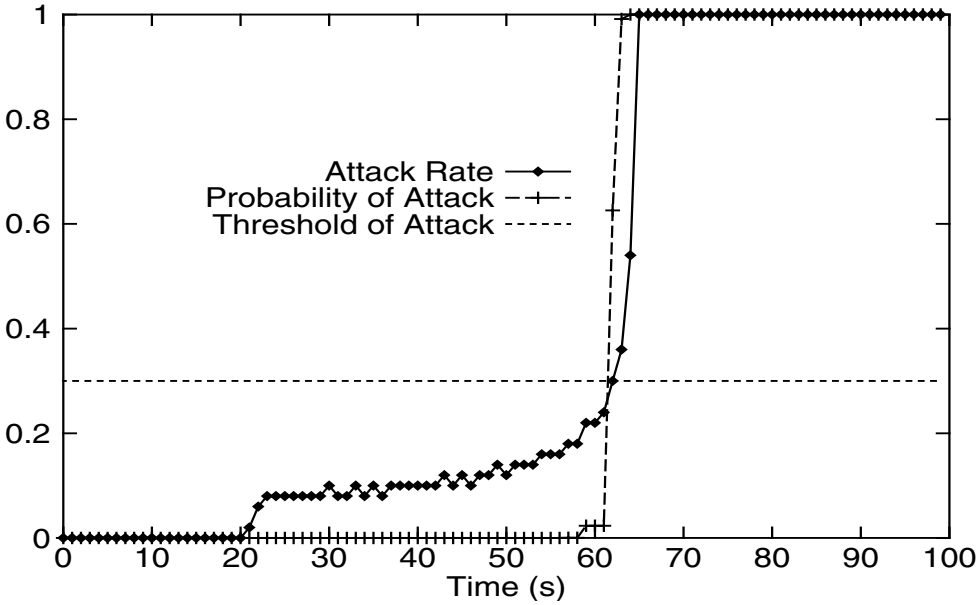


Figure 10. Attack Rate and Attack Probability

Figure 10 illustrates the application of the results of the cluster analysis to the on-line detection of network attacks. Here, the established classification rules are used to compute a sequence of events (4.1) based on the varying status of the network. These events are used to update the probability of attack. In the figure, time is shown in seconds and the reference time was arbitrarily chosen just prior to the start of the attack. The figure presents both the rate of attack packets entering the network, normalized to the maximum rate observed, and the probability of attack iteratively calculated as per (4.3).

This figure shows that the attack begins gradually, at about time 20sec, and increases in an exponential manner until it reaches its maximum rate. During the initial stages of the attack, the incoming traffic is light and does not have a significant effect on the probability of attack. At about time 58sec, the attack begins to increase in severity, and the probability of attack reacts quickly. Based on this data, we have set the "threshold of attack" at .3, at which point notification should be made in order that countermeasures be taken. It is interesting to note that the probability of attack reaches 1 before the attack reaches it full force, leading the attack and providing early warning. It should also be noted that the typical duration of a TCP SYN flood is measured in hours, while the proposed approach detected the attack in less that 40 s. The actual detection time is based on both the rate of progress of the attack and the normal traffic patterns on the network being protected.

Ping Flood Attack

The results of cluster analysis for Ping Flood on the experimental network are shown in Figs. 11.1-11.3. The informative components of the network status chosen by the cluster analysis procedure are fraction of IP control packets (x_{15}), fraction of control packets (x_{17}), rate of change in the fraction of IP control packets (x_{16}), and fraction of Large packets (x_5). These were combined into the informative subspaces (x_{17},x_{15}), (x_{16},x_{15}), and (x_{15},x_5). Ping Flood attacks are characterized by a large number of IP control packets (ICMP echo packets) flooding the target network. These packets are generally large.

From Figure11.1, it is clear that the most informative characteristics of a Ping flood attack are a large fraction of control packets, nearly all of which are IP control packets, consistent with the nature of the attack. Figure 11.2 shows that ping flood attacks can be distinguished from normal traffic that has a high fraction of IP control traffic based on the rate of change of IP control traffic. Figure 11.3 also illustrates that normal traffic can have a wide range of fraction of IP control traffic, but Ping Flood attacks can be distinguished by their high fraction of large packets in the network.

Figure 12 illustrates the application of the results of the cluster analysis to the on-line detection of ping flood attacks in the target network. This figure shows that the attack begins gradually, at about time 30s, and increases in an exponential manner until it reaches its maximum rate. Like the TCP SYN flood attack, during the initial stages of the attack, the incoming traffic is light and does not have a significant effect on the probability of attack. At about time 80s, the attack begins to increase sharply, and the probability of attack reacts. Again, the probability of attack leads the attack itself, reaching 1 before the attack reaches full force. Also, the attack was detected very early, in about 50 s in this experiment. Actual attacks are measured on a scale of hours.

It could be seen that an attack warning message could be generated should the attack probability value exceeds certain threshold. Otherwise, application of the developed software provides the network operator some important insights.

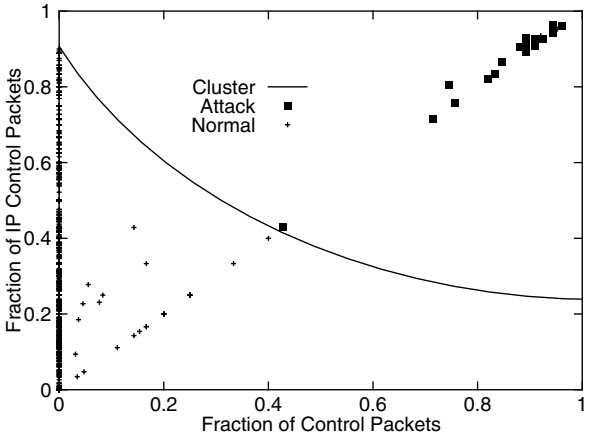


Figure 11.1

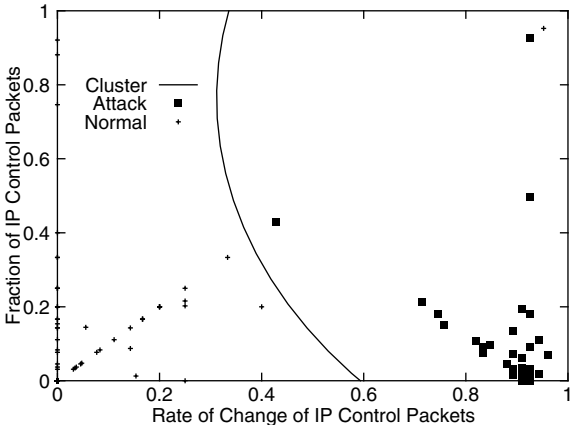


Figure 11.2

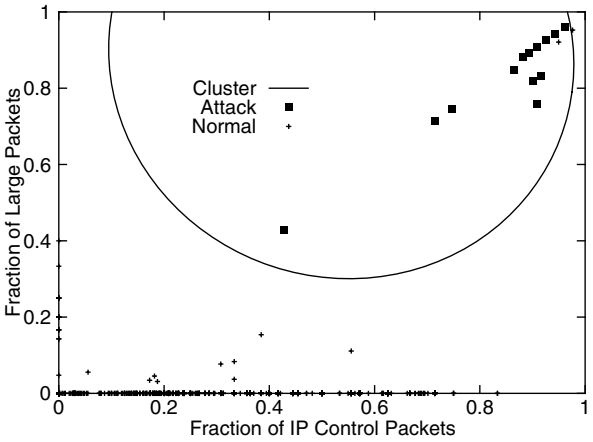


Figure 11.3

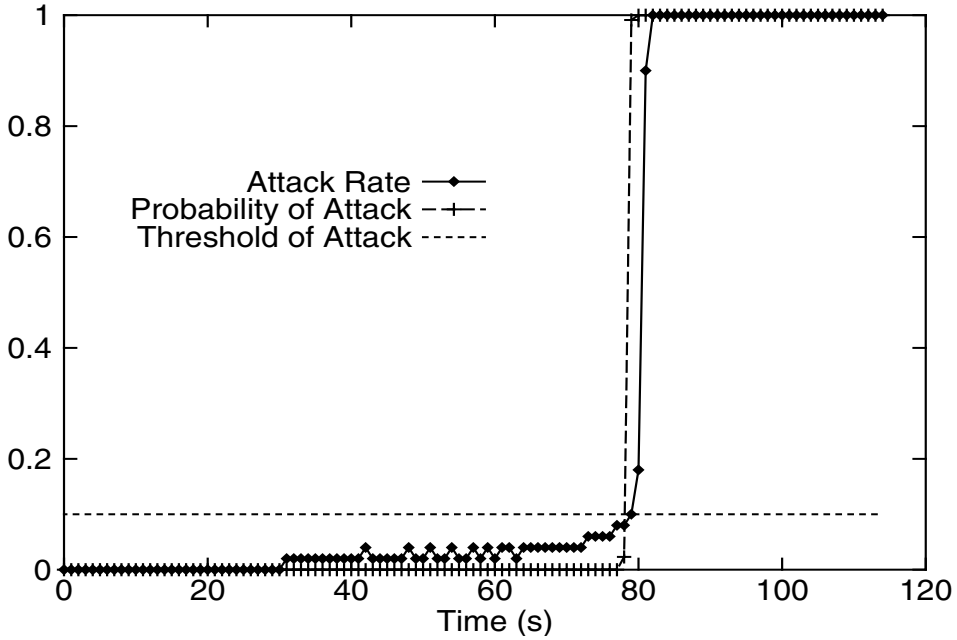


Figure 12. Attack Rate and Attack Probability

7. References

1. V. Skormin and C. Herman, “Application of Statistical Clustering for Process Control of Screen Printing”, ASME Electronic Packaging Journal, September 1995

2. V. Skormin, L. Popyack and V. Gorodetski, “Data Mining Technology for Failure Prognostics of Avionics”, IEEE Transactions on Aerospace and Electronic Systems, will appear in April 2002

3. M. D. Vose, “The Simple Genetic Algorithm: Foundation and Theory,” *MIT Press*, Cambridge, MA, 1999

4. V. Skormin, V. Nikulin, and T. Busch, “Application of Genetic Algorithms for Optimal Design of Acousto-Optic Beam Steering Components,” Optical Engineering, will appear in January 2002

Click here to view PowerPoint presentation; Press Esc to exit

Network Mapping Tool for Real-Time Security Analysis

Frédéric Massicotte, Tara Whalen, and Claude Bilodeau
Communications Research Centre Canada (CRC)
3701 Carling Avenue
Ottawa, Ontario, K2H 8S2
Canada

{frederic.massicotte, tara.whalen, claude.bilodeau}@crc.ca

Abstract

This paper introduces a prototype network mapping tool that can be used along with intrusion detection systems to provide, in real-time, a comprehensive picture of network topology. This software tool can generate descriptions for both physical and logical connectivity of network components. It also provides positive identification of the operating systems running on the networked machines, as well as state and configuration information about the hosts and their connectivity.

The mapping of a network is performed by following a series of automated steps, that use a number of elements to query network components: built-in networking protocols (ICMP, ARP, NetBIOS, DNS), standardized management protocols (SNMP), freely-available mapping tools (nmap, Xprobe), and a number of CRC-developed intelligence databases and programs for analysing the results returned from queries. The artificial intelligence component takes a set of possible paths and creates a full network map. A network monitoring component updates a database of connections between machines and displays current and past communications links on a graphical interface. It is also able to determine when a new machine has been added to the network, which is a vital part of updating the network map. Integrating these information-gathering tools makes network mapping highly accurate and up-to-date, allowing for real-time analysis of attacks and changes in topology.

1 Introduction

Because of the complexity of networks and attacks on networks, analysts require as much information as possible about their own network infrastructure. Knowledge of host connectivity and equipment configuration is essential when one must investigate or react to computer or network attacks. In particular, managing network security is almost impossible without knowledge of network topology. Deploying network sensors and defensive tools, such as firewalls, cannot be done effectively without knowing where these components must be positioned.

Furthermore, existing Intrusion Detection Systems (IDSs) can detect a number of attacks, but they also generate false positives. It would be therefore helpful to provide additional information that can assist an analyst in correctly determining the nature of an attack (or concluding that no such attack is in progress): for example, in indicating the likelihood that an attack is genuine or possibly the result of a misconfiguration. One of the problems with current IDSs is that they do not provide sufficient information about the network they monitor. They do not present the "big picture" of what is going on in a complex network. It is beneficial to provide information such as: the type of operating system on each host, what types of network services are offered, and how these computers are interconnected. As well, a dynamic map is necessary to reflect changes in the network as they occur in real time.

CRC has developed a prototype tool that provides this type of information to a network analyst via a graphical user interface. This tool illustrates connectivity among network components, including the type of links, and also reveals the operating systems used by the network components and the role that they carry out

Paper presented at the RTO IST Symposium on "Real Time Intrusion Detection", held in Estoril, Portugal from 27-28 May 2002, and published in RTO-MP-101.

(e.g. router, workstation). There exist other tools that perform similar tasks, but a survey conducted in-house found them to be either incomplete or inaccurate. By combining a number of applications and information sources, one is able to construct a comprehensive and trustworthy picture of a network.

The next section discusses the motivation for creating this prototype tool. Following this is a description of the components used by the tool, along with an overview of how these are used to build a map. After this overview, the reader is presented with an in-depth description of how the mapping tool was implemented, including a visual example of an actual map. Finally, other applications for the tool and plans for future development work are discussed.

2 Motivation

A network mapping tool is a natural complement to intrusion detection systems. Analysts who need to make rapid response decisions need as much information as possible about the network so that attack information can be placed into perspective. A major limitation of IDSs that can be alleviated by network mapping is false positives, in which the IDS claims that an attack is taking place when in fact no such attack is occurring. Dealing with this problem requires a spatial and temporal *context* for the attack, so that an analyst can more easily determine whether such an attack is real; for example, a false attack report may actually be due to a misconfiguration or harmless yet unusual activity between two hosts. Network mapping provides this context, through visually presenting network topology and communications. Furthermore, dynamic network maps can reveal violations of network security policy, such as a wireless network being added in a secure area where wireless communications are forbidden. Network mapping is a very useful method for securely managing complex networks, particularly those that are very dynamic or heterogeneous, such as a coalition network.

CRC recently analysed the most popular IDSs and network mapping tools available today. It was found that all have limitations that prevent the creation of a complete, accurate network map. Ideally, a mapping tool should have the capability to provide: a comprehensive map that spans several segments (i.e., able to show hosts separated by routers), detailed levels of host connectivity (physical and logical topology), identification of every network component (hosts, routers, switches) and state and configuration information relating to the software and services on each machine (e.g., operating system (OS)). Such a tool should also provide dynamic updates of this information.

Six of the major mapping tools surveyed by CRC are: Ipswitch Inc. What's Up Gold 6.02, Network Associates Cybercop Scanner 5.5, Cheops-ng 0.1.5, Fluke Networks LAN Mapshot 1.5, Microsoft Visio Enterprise Network Tools 2002, and HP OpenView Network Node Manager 6.2 [1-6]. Only HP OpenView and Fluke LAN Mapshot could generate an automatically-updated map. HP OpenView cannot map beyond the local segment, and is also very restricted in which operating systems it can correctly detect. Similarly, Fluke LAN Mapshot cannot map beyond its local segment and cannot display both routers and hosts on the same map. Among the six tools, only Microsoft Visio can map a domain beyond its local segment, but its map was very incomplete (missing hosts) and does not support OS detection. The other three tools mentioned provide only limited (or non-existent) support for OS detection, and often display inaccurate or incomplete results. The tool proposed in this paper attempts to remedy these deficiencies.

3 Mapping Tool: Information Gathering Methods

A number of software components are used to query workstations, servers, and network components: built-in networking protocols (e.g., ICMP, ARP), standardized management protocols (SNMP), freely-available tools (e.g., nmap, Xprobe), as well as CRC-developed probing tools, databases, and programs to analyse the results returned from these queries. The majority of the network information is gathered by using components of the TCP/IP protocol suite. Some of these mechanisms are briefly described below; for more information, the reader is directed to Stevens [7].

TCP/IP: different implementations of the Transmission Control Protocol/Internet Protocol (TCP/IP) stack may respond slightly differently to network probes[1]. The small differences in response to these probes can indicate what specific operating system is running on a networked computer. This technique is sometimes called "OS fingerprinting" and is used by such applications as nmap ("network mapper") [8-9], a tool designed to scan networked systems for security auditing or simply for network investigation. As well as performing OS fingerprinting, nmap also tests a number of ports to see what networked services respond. This reveals what services are running and which ports are open or closed. Through the use of nmap, CRC's tool can provide information about software running on the network, including operating systems and servers.

ICMP: the Internet Control Message Protocol (ICMP) uses several message types that assist TCP/IP communications. For example, it can be used to determine whether a machine is alive on the network. An ICMP echo request is sent to an IP address (e.g., through *ping*), and if there is a machine at that address, it will send back an ICMP echo reply. The *traceroute* tool also uses ICMP to reveal a particular hop-by-hop path between two hosts. As well, the third-party tool Xprobe [10-11] takes advantage of minor variations in ICMP implementations to determine what operating system is running on a networked host, based on specific replies to ICMP probes.

DNS: the Domain Name Service (DNS) provides a mapping between numeric IP addresses and textual hostnames, via a distributed database. An application can use DNS to find the name of a host from its IP address or vice-versa, by contacting the name server. This allows the prototype tool to provide hostname labels on the topological map for the machines discovered within a range of IP addresses.

ARP: the Address Resolution Protocol (ARP) is used to map an IP address to a physical address (i.e., of a network card). ARP sends out a broadcast message asking for the media access control (MAC) address —the physical address—associated with a specific IP address. The machine matching that IP address sends back its MAC address. ARP can thus be used to fill in MAC addresses in the map for all machines on the local segment. Furthermore, any machines added to the network segment will send ARP broadcasts when they start to communicate: passive monitoring of these messages allows the prototype tool to quickly and easily update the network map.

SNMP: the Simple Network Management Protocol (SNMP) is used in numerous tools for managing network components. Information about systems, performance, etc., is stored in Management Information Bases (MIBs); these MIBs can be queried to provide vast amounts of data (depending on which MIBs are supported), such as information on routing and operating systems.

NetBIOS: the Network Basic Input/Output System (NetBIOS) is a protocol for supporting services over local area networks [12]. It facilitates inter-application communications and data transfer (such as file sharing). NetBIOS can be used to discover resources offered by a networked server, and can indicate what OS is running on a host.

4 Mapping Tool: Creating the Network Map

The tools and protocols described in Section 3 are used in conjunction with the tool's own software to create a dynamic network map. The mapping takes place in three phases: first, network and host information is gathered and an initial map is drawn; second, a search for new devices begins by passively monitoring the network traffic; third, if a new device is found, further information is gathered about this new component, and the map is updated. Once phase three is completed, the tool resumes its operation by returning to phase two.

[1] Commonly, a specially crafted IP or ICMP packet sent to a node to gather configuration information.

4.1 Phase One: Initial Mapping

The prototype mapping tool works in an IP network: it is given a range of IP addresses to map, and proceeds to gather network and host information within that range. It is worth noting that the tool can run on any workstation within the designated IP range: because of the use of standard tools and protocols, the mapping software can be easily deployed anywhere in a network. Once initiated, the mapping tool proceeds in five distinct steps to gather its data. These steps form the basis of the protocol tool; they are described briefly in this section and are detailed later in the paper.

- **Host Discovery:** This step aims to find every active or shut-down computer that currently is, or has recently been, part of the network. To accomplish this task, several different scanning approaches are used (e.g., ICMP echo reply, ARP).

- **OS Discovery:** Once the devices that are part of the network are found, the tool tries to identify which operating system they are running, the role the device carries out (e.g., router, workstation), and the manufacturer name for networking devices.

- **Resource Discovery:** This step finds out the services and shared resources offered by every computer located on the network.

- **Connectivity Discovery:** Once the available resources and the services offered are identified, the tool uncovers how the hosts are connected together at the physical layer and at the network layer.

- **Identity Discovery:** Finally, the tool collects the identity of all the networked computers. Identity information includes the host's DNS name, NetBIOS name, and MAC address.

4.2 Phase Two: Listening for Updates

Once the initial mapping is complete, the real-time, dynamic phase begins. A sniffer is activated to find any computers that have been added to the network since the mapping began. When a new host attempts to communicate on the network, it will usually broadcast an ARP request to find the hardware address of the machine to which data will be sent. These broadcasts are detected by the tool's sniffer, and the map will then be updated to reflect this new device.

4.3 Phase Three: Updating Network Map

When a new device is found, its connectivity, state and configuration information must be added to the map. The tool repeats the last four steps, outlined in Section 4.1, for the new machine: that is, the operating system, resources, connectivity, and identity information is gathered for the newly-discovered host and the new map is generated in real time. The tool then returns to the listening phase, waiting to see if any new machines are added. This method ensures that the network map remains current.

5 Software Implementation Details

The network mapping steps require the integration of a wide variety of tools, including using automated logic to generate a correct map. The software implementation is described in detail in this section. The mapping tool itself runs on a network workstation, with the master code running on Microsoft Windows and additional functionality provided by Linux libraries. This is accomplished by installing the tool on a workstation that runs VMware, which is an application program that allows several OSs to co-exist and share the same physical environment. The user specifies a range of IP addresses to map, and whether a physical view or logical view (also called *network view*) is preferred.

5.1 Host Discovery

In the first phase of information gathering, a wide range of techniques is used to gather information about currently-active and previously-active hosts located on a network:

- **ICMP echo request:** ICMP echo request messages are sent across the IP range to determine which addresses are valid, based on whether or not a response is returned.

- **DNS zone transfer queries:** The name server is accessed for a *zone transfer*, which is a request for the entire DNS database for the domain. The tool obtains all the registered IP/hostname pairs available on the network. If this transfer request fails (e.g., due to problems with DNS implementations), the tool will try to send individual DNS queries for any address not responding to an ICMP echo reply or ARP request. This is a way to determine whether the machine is temporarily down, or has never been active: if the DNS has no entry for an IP address, then that address is likely invalid, rather than simply currently unreachable.

- **ARP requests:** ARP requests are sent to local machines (within the broadcast domain) in order to find the MAC addresses corresponding to IP addresses. This host-finding protocol is also used as a passive method for finding any new computers that connect to the network after the initial mapping phase has completed.

- **SNMP requests:** SNMP MIBs provide large amounts of information about systems, and are particularly useful for discovering machines that are temporarily down or disconnected. The SNMP ARP MIB (when supported) will maintain information about the IP/MAC mapping for hosts and can be used to implicitly detect machines that did not respond to direct queries.

5.2 OS Discovery

The next step is to uncover as much information as possible about the operating systems running on the hosts that were found during the previous querying step. As well as pinpointing the operating system, the tool must determine the *type* of device (i.e., the role that it plays in the network), so that it can display workstations, switches, and routers, for example, as different icons in the topological map. The tool is able to discover the function and OS of the devices through the following methods:

- **OS fingerprinting:** The third-party tools nmap and Xprobe are used to help determine what OS each host is running. As described above, nmap uses IP, TCP, UDP, and ICMP probes to pinpoint an OS type and version. Xprobe works in a similar fashion, but uses only ICMP probes. Xprobe has been modified slightly to allow for more accurate results in determining the OS.
- **Banners:** Networked servers expose banners that are another valuable source of operating system information. Quite often, connecting to a service such as *telnet* will result in being shown a banner that announces explicit operating system information, or otherwise presents information that indicates a category of operating systems. CRC developed a banner grabber tool that analyses the banner (using string matching) to determine if this information can be used in determining the operating system. The tool attempts to gather banner information from several sources, including telnet, ftp, ssh, and http.
- **SNMP and NetBIOS:** The tool also generates explicit queries for OS information available through SNMP and NetBIOS. SNMP MIBs, both standardized MIBs as well as proprietary MIBs created for specific products, are a valuable source of information. For example, the standardized MIB for management of TCP/IP-based internets [13] supports the variable sysObjectID. The latter can be associated with a specific hardware component (like a network switch from a specific company) or associated with a specific operating system. The proprietary MIBs are useful for uncovering OS information for networking devices: Cisco, for example, provides a MIB that holds the exact version of the Internet Operating System (IOS) running on a router. In addition to SNMP, NetBIOS can be used to discover a Windows operating system running on a host, if administrator access is available.

However, the information returned from these queries is often inaccurate, so the tool needs to combine query results to increase accuracy. To illustrate the mechanism, let us compare the query results returned when probing a Windows 98 host and a Windows NT host, as shown in Table 1.

Table 1: Operating system query example

query method	target OS: Windows 98 **OS query result**	target OS: Windows NT **OS query result**
nmap	Windows 95 or 98 or NT	Windows 95 or 98 or NT
Xprobe	not Windows 95	not Windows 95
NetBIOS	Windows 95 or 98 or ME	Windows NT
	conclusion: Windows 98	**conclusion**: Windows NT

When nmap queries a Windows 95, 98, or NT machine, it will always return the result "Windows 95/98/NT" because it is unable to differentiate between these systems. If XProbe performs a query on the same machine, it will return "Windows 95," if that OS is in fact running. Otherwise, its conclusions are often incorrect. NetBIOS can identify NT systems but cannot discriminate between Windows 95, 98 or ME systems.

Performing this integrated analysis requires re-formatting the different query results and looking for a consistent OS among the aggregated results. To derive a clear result and ease the procedure, CRC has developed databases that generate standardized output based on the different third-party tool results. Because each tool presents different sets of data with different formats, these must be reformatted so that the data may be integrated and ready for comparison. Looking for the consistent OS consists in finding a common intersection among the normalised results: if the data consistently indicates one operating system, there is a perfect match; else, the set of conflicting operating systems is returned with the conclusion "unknown." When a match is found, the map can be updated with the icons for operating system type as well as for machine type (e.g., workstation or router).

5.3 Resource Discovery

Once the operating systems have been determined, the next step is to find networked resources offered by each computer in the network. To find these resources, a port scanner is used in conjunction with NetBIOS:

- **Port Scanning:** To discover open TCP and UDP ports, the tool relies on port scanning tools, predominately nmap. This procedure can discover if a specific port is open, closed, or possibly has its access filtered by a firewall. The tool then uses CRC's own database to map ports to services commonly offered on those ports, thus revealing what networked services are likely being hosted on a workstation.
- **NetBIOS:** Networked Windows hosts often share drives; using NetBIOS, the tool can effectively discover hidden and public shared drives.

5.4 Connectivity Discovery

At this stage, the tool has some idea of what connections exist, but still needs to create a consistent picture of links. To do this, an artificial intelligence (AI) program takes the set of possible paths and creates a full network map. An AI approach was chosen in order to effectively process the rule sets that CRC developed for creating a consistent map. The map that the AI component derives is based on partial information, so it is not complete: the derived set of links is a subset of the true set. In practice, the program has been successful on small networks, and has been effective in updating the map with new hosts

The map can be created for two topology types: logical (network level) and physical (physical link-level). (A session-level map that will represent TCP sessions is currently being developed.) The physical link-level map includes low-level information, such as specific port connections on network devices. Conversely, the network-level map shows a more high-level view of the topology, in terms of segments and routers. To create these two maps the following techniques and protocols are used:

- **Network level:** To draw network links between two different networks, tools such as traceroute are used to find routers, and the route between the networks is deduced. SNMP is also used to request the routing table from each router. This allows the tool to draw a more precise picture of the network, because the management information is beneficial in revealing local topology.

- **Physical level:** To draw the map at the physical link layer, the tool currently relies on SNMP, querying the bridge MIB wherever it is supported in a networking device. An AI algorithm is then used to process this partial information and create as complete a map as possible.

5.5 Identity Discovery

Once connectivity discovery has finished, the tool completes the identity information for each host. Host identity includes information such as MAC address, DNS name, and NetBIOS name. At this stage, the tool has already gathered this information for all live hosts on the local segment. For remote hosts and those that are temporarily down, it must make further queries through DNS and SNMP to fill in the identity information.

5.6 Additional Information

In addition to the information presented so far in this section, the tool also gathers a variety of data that can be used for network security management. This includes the state of each host that was discovered (active or down), the speed of each link in megabytes, the type of link (e.g., wireless, serial, Ethernet), and the users listed in the registry of each host.

6 Examples of Network Maps

Figures 3.1 and 3.2 show two screenshots, taken from maps of a testbed, that show the topology of diverse network components. Figure 3.1 contains a display of the entire network whereas Figure 3.2 highlights the kind of information that the tool can gather automatically about a specific host.

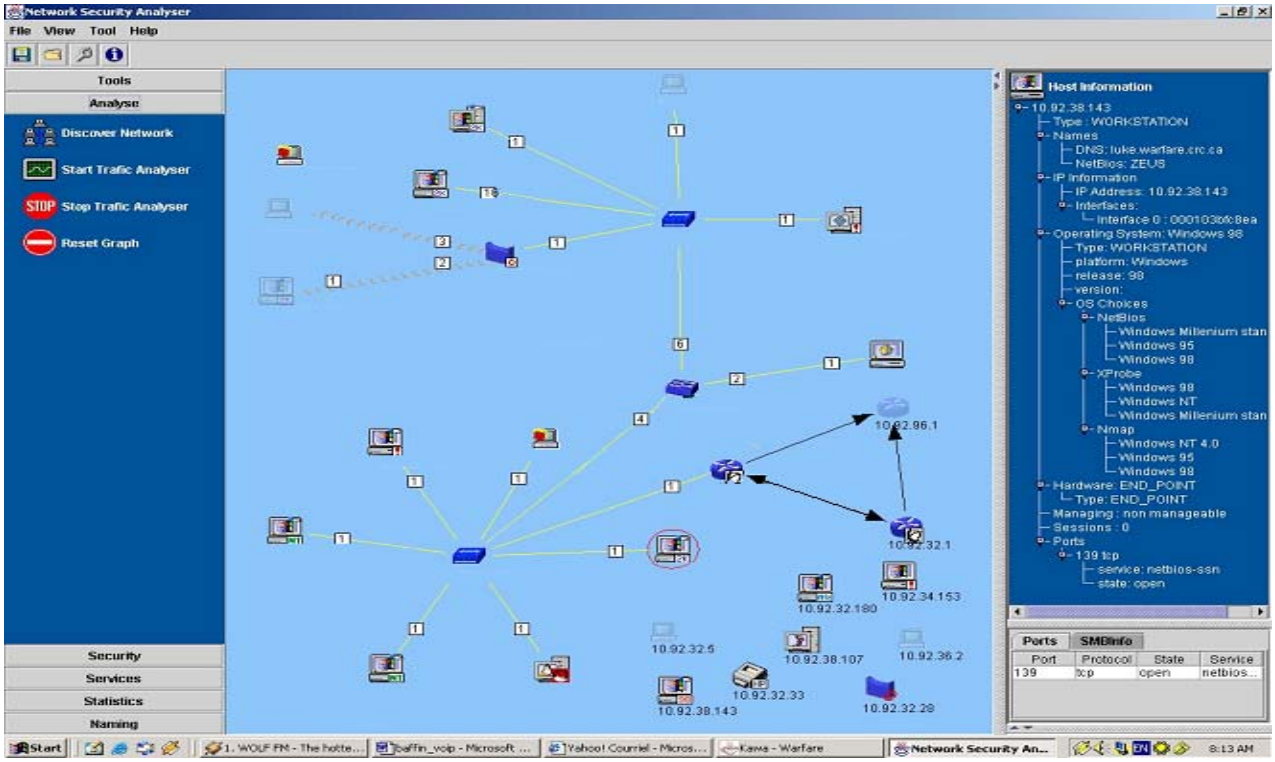


Figure 3.1 Network map of a test network as generated by the prototype tool

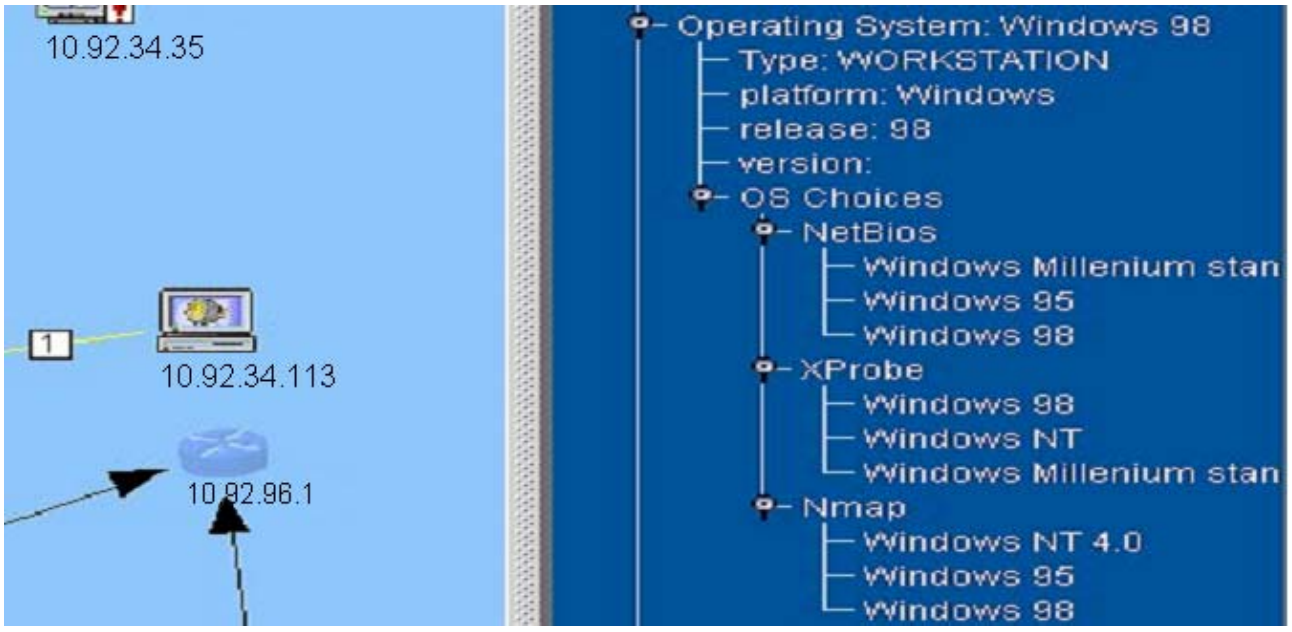


Figure 3.2 Detailed host information displayed on network map

7 Related Applications and Future Work

The prototype tool described in this paper can be used as a plug-in for many different security management systems. It can form the foundation of network analysis programs, such as traffic visualizers, data communications visualizers, or event correlators. It can be also used with a network management tool to provide a real-time, graphical view of the network, which would be valuable for securely managing a diverse coalition network. However, the major strength of this tool resides in its security aspect. It provides a solution for creating an accurate picture of a network. Used as a graphical plug-in for vulnerability detection systems, it provides a means for displaying host vulnerabilities along with basic system configuration, link state information and network topology.

Being a prototype, this tool currently has some limitations that will need to be addressed in future development. For instance, the tool currently cannot detect network routing loops, discover devices that do not possess an IP address, or automatically determine when anomalous network behaviour occurs. Also, the module that updates the map dynamically is still in development.

With regard to future work, the principal focus will be to investigate entirely passive methods for mapping the network and gathering network information. This method is advantageous, as it consumes fewer network resources, and is less easily detected by attackers. It will be desirable to see if the tool can be combined with tools that perform traffic modeling, network management, or knowledge management. In the area of traffic modelling, CRC has begun to explore using this tool as part of a traffic summary agent, which could summarize large amounts of traffic between computers. This would allow a network administrator to quickly identify problems or unusual behaviour in the network by using the visual representation of traffic shown by the map.

For knowledge management, the authors plan to use this tool to help IDSs to detect attacks on a network. It is hoped that the tool can be further improved to make it a fast, real-time network *information* tool that incorporates a knowledge base of network components. An IDS could then query the knowledge base for information about each component in the network. The tool will be used to correlate information between different types of security tools, such as firewalls and IDSs, to give to a security administrator a more precise picture of the network. To add this functionality, it will be necessary to investigate the addition of correlation algorithms to the prototype.

8 Conclusions

This paper has presented a software prototype of a new network mapping tool that assists network monitoring, providing accurate information and detailed maps. The tool is versatile and can be used in a variety of security applications. Several projects are underway to develop it further, including the integration of this tool with intrusion detection systems and vulnerability detection systems in order to support real-time network anomaly display, and integration with traffic visualization systems.

9 Acknowledgments

This work was funded by Defence R&D Canada, Department of National Defence.

10 References

[1] Ipswitch Inc.'s WhatsUp Gold. http://www.ipswitch.com/Products/network-management.html

[2] Network Associates CyberCop Scanner. http://www.cybercop.co.uk/cybercop/scanner/default.htm

[3] Cheops-ng. http://cheops-ng.sourceforge.net

[4] Fluke Networks LAN Mapshot. http://www.flukenetworks.com/us/LAN/Monitoring+Analysis+Diagramming/LAN+MapShot/Overview.htm

[5] Microsoft Visio Enterprise Network Tools. http://www.microsoft.com/office/visio/evaluation/indepth/network.asp

[6] HP OpenView Network Node Manager. http://www.openview.hp.com/products/nnm/index.asp

[7] Stevens, W. Richard. TCP/IP Illustrated: the protocols. Addison-Wesley, 1994.

[8] Fyodor. nmap-254BETA29. http://www.insecure.org/nmap

[9] Fyodor. *Remote OS Detection via TCP/IP Stack Fingerprinting*. October 1998. www.insecure.org/nmap/nmap-fingerprinting-article.html

[10] Arkin, Ofir. Xprobe. http://www.sys-security.com/html/projects/X.html

[11] Arkin, Ofir. *X - Remote ICMP Based Fingerprint Techniques.* August 2001. http://www.sys-security.com/archive/papers/X_v1.0.pdf

[12] NeonSurge. *Understanding NetBIOS*. http://www.signaltonoise.net/library/netbios.htm

[13] McCloghrie, K., and Rose, M. *Management Information Base for Network Management of TCP/IP-based internets: MIB-II*. RFC 1213. March 1991.

Cliquez ici pour voir la présentation PowerPoint; appuyez sur la touche ESC pour sortir

Fonction de réaction (Reaction Function)

Mamadou Diop
ALCATEL CIT
1, rue Ampère
B.P. 56
91302 Massy
France

Mamadou.Diop@alcatel.fr

Sylvain Gombault
ENST-Bretagne
2, rue de la Chataigneraie
B.P. 78
35512 Cesson-Sevigne Cedex
France

Sylvain.Gombault@enst-bretagne.fr

Résumé : MIRADOR[1] est un Programme d'Etude Amont (PEA) lancé par la DGA/CELAR/CASSI, visant à réaliser une plate-forme coopérative et évolutive d'outils de détection d'intrusion, dotée de facultés de réaction. Alcatel en assure la maîtrise d'œuvre. Après avoir rappelé quelques définitions, cet article décrit la fonction de réaction étudiée dans le cadre du PEA MIRADOR. Nous proposons une taxonomie des actions composant la réaction et montrons comment les différents paramètres de cette taxonomie sont utilisés pour fournir une fonction de réaction fiable, flexible et intégrée à l'architecture de Mirador. Par exemple, nous définissons pour chaque action un rôle pris parmi 4 : Information, Dissuasion, Correction ou Compensation. Ce rôle nous permet de gérer l'état de vulnérabilité d'un système qui a fait l'objet d'une intrusion. Nous présentons ensuite les éléments de configuration de cette fonction de réaction et décrivons l'algorithme de lancement des différentes actions en tenant compte de leur rôle et de leur mode de déclenchement (automatisé ou manuel). Nous terminons en présentant l'ergonomie de l'interface graphique offerte à l'administrateur de sécurité.

1 Introduction

Les mécanismes de réaction ont été enrichis dans les différents IDS (Intrusion Detection System) qui proposent désormais, en plus des alarmes et rapports, des actions élaborées. Cependant, on s'aperçoit rapidement qu'il n'existe pas une grande variété des techniques de réaction d'un outil à l'autre et que les critères de décision qui leur sont associés restent souvent simples voire simplistes.

De plus, dans un contexte d'exploitation, on remarque que les actions qui paraissent les plus intéressantes comme la reconfiguration automatique de garde-barrière ou de routeur ne sont pas utilisées dans la pratique par manque de confiance dans les capacités de l'outil à réagir ou à l'inverse, par peur de ne pas maîtriser les conséquences d'une automatisation des contre-mesures [1].

Enfin, la majorité des actions proposées sont des parades ayant pour but de stopper l'attaque en cours. Les actions plus élaborées ayant pour fonction de corriger automatiquement les vulnérabilités détectées, restent marginales.

Après voir rappelé quelques définitions, ce document décrit la fonction de réaction dans le cadre du PEA MIRADOR. Son objectif est de fournir une fonction de réaction fiable et efficace dans laquelle un opérateur de sécurité aura toute confiance.

2 Définitions

Réaction/(ré)actions

Nous définissons la réaction comme étant l'ensemble des actions menées par un site suite à la détection d'une intrusion.

[1] MIRADOR : Mécanismes de Détection d'Intrusion et de Réaction aux Attaques en Domaine militaiRe

Communication présentée lors du symposium RTO IST sur « La détection des intrusions en temps réel », organisé à Estoril, au Portugal, 27-28 mai 2002, et éditée dans RTO-MP-101.

Nous utilisons le terme ***réaction*** toujours au singulier, en référence à la ***fonction de réaction*** d'un IDS. Nous n'employons pas la forme plurielle (réactions) qui représenterait l'ensemble des actions en réaction à une attaque donnée, nous lui préférons le terme ***actions*** qui évite toute ambiguïté.

Site

Un ***site*** est un ensemble d'équipements informatiques, reliés en réseau, auquel s'applique une même politique de sécurité. Un IDS surveille les équipements (réseaux et systèmes) de l'ensemble du site.

Service

Un ***site*** donné offre un ***service*** en interne ou en externe, qui est sa raison d'être. Ce service peut englober des services élémentaires tels que le partage de fichier, le web ou la messagerie. Nous considérons qu'il y a ***dégradation*** du service offert par le site lorsque, pour des raisons de sécurité, la fermeture de ports utiles pour ce service s'avère nécessaire.

3 Taxonomie des actions

Nous proposons une taxonomie des actions composant la réaction et utilisons les différents paramètres de cette taxonomie pour enrichir la fonction de réaction décrite au paragraphe 5.

Effet

Une action a un effet ***actif*** sur une intrusion si elle a pour objectif de stopper l'attaque, donc de diminuer ses effets sur les ressources du site et, si possible, de corriger la faille utilisée. Il peut s'agir d'une coupure de communication.

Une réaction **passive** n'agit pas directement sur l'intrusion, c'est par exemple une alarme prévenant un opérateur ou le déclenchement d'une collecte d'informations.

Remarque : une contre-mesure peut être active ou passive.

Type

Une ***action*** peut être de deux types :

- une ***alarme*** qui se présente sous la forme d'un événement envoyé à destination d'un administrateur de sécurité, éventuellement au travers d'autres outils d'analyse de plus haut niveau,
- une ***contre-mesure*** a un effet actif sur l'attaque et l'attaquant et a pour but de changer son comportement actuel et/ou futur.

Rôle

Nous associons à chaque action un ***rôle*** pris parmi 4 : ***information***, ***dissuasion***, ***correction*** ou ***compensation***.

- Les alarmes, rapports, captures de paquets ont un rôle d***'information***. Ce rôle comprend à la fois l'envoi et la collecte d'informations.
- Les actions de ***dissuasion*** n'ont d'effet que sur l'attaquant et doivent être proportionnelles à la gravité de l'intrusion. Cela peut aller du ralentissement des acquittements jusqu'à la provocation d'un déni de service sur la machine attaquante.
- Les actions de ***compensation*** agissent sur l'attaque en cours en la neutralisant. Cependant, soit elles laissent le système dans un état vulnérable, soit elles conduisent à une perte de ***service***.
- Enfin, celles de ***correction*** ont une action sur la vulnérabilité utilisée par l'attaque. Elles corrigent la politique de sécurité du site et ont donc une action permanente, sans remettre en cause le ***service*** fourni par le ***site***.

Nous donnons au paragraphe 6 des exemples d'actions pour chacun de ces rôles.

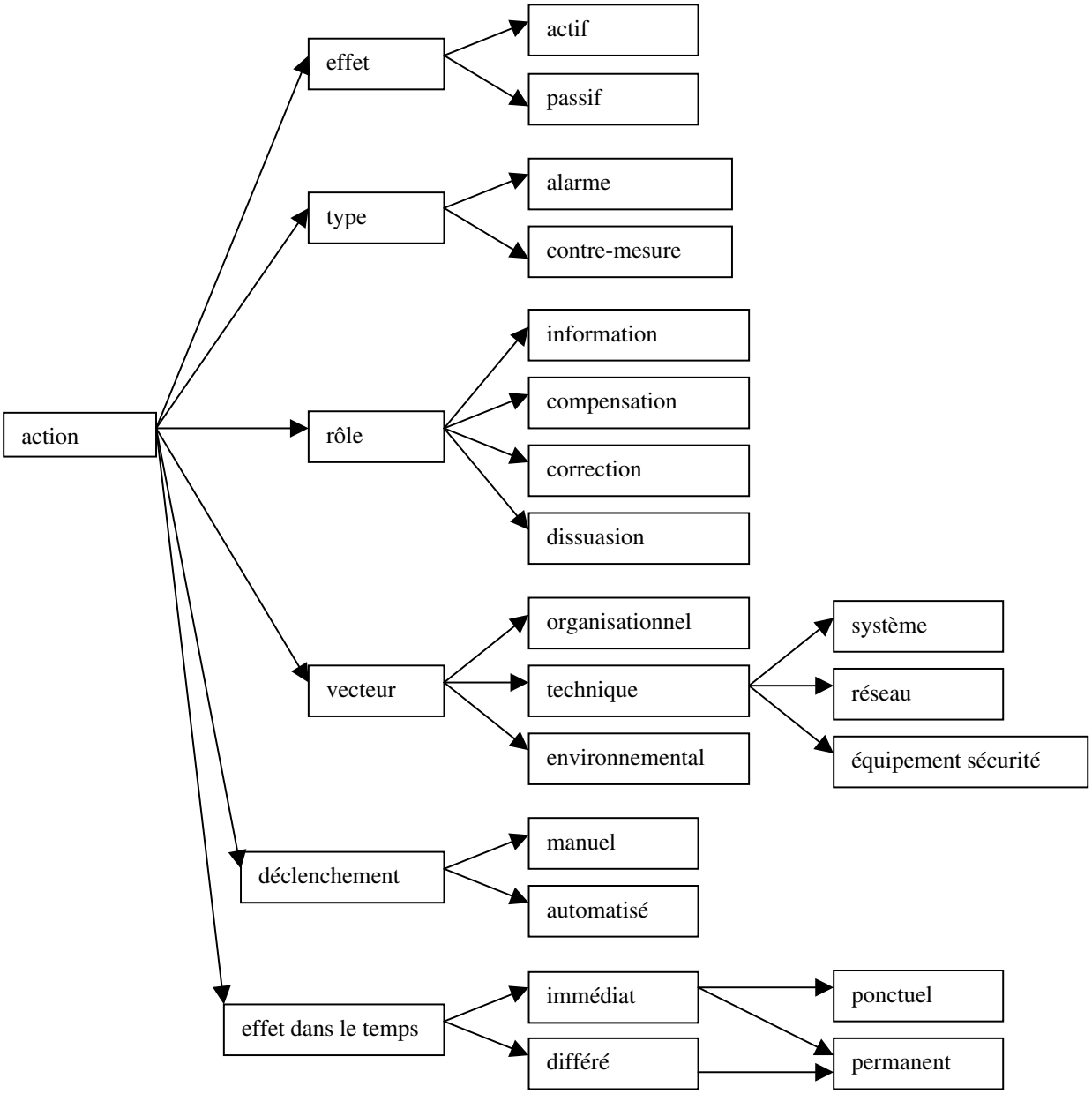


Figure 1. Taxonomie des actions

Vecteur

Le vecteur est le moyen grçace auquel l'IDS exécute une contre-mesure. Le vecteur peut être ***technique***, ***organisationnel*** ou ***environnemental***. Les vecteur techniques identifiés sont : le ***réseau***, l'***équipement sécurité*** et le ***système***.

Pour réaliser une action au travers des vecteurs système et équipements de sécurité, la méthode consiste à s'y connecter avec les droits de super-utilisateur et à lancer les commandes appropriées.

Pour le vecteur réseau, le problème est moins simple : de nombreux IDS tels que NePoSe, eTrust ou Snort offrent la possibilité d'utiliser le réseau comme vecteur d'action pour générer du trafic, mais cette fonction n'est pas toujours accessible de l'extérieur et la fonction de réaction ne peut y avoir accès directement.

Déclenchement

Le ***déclenchement*** d'une action peut être ***automatisé*** ou ***manuel***. Dans ce dernier cas, l'opérateur a la possibilité de valider les contre-mesures avant leur exécution.

Effet dans le temps

Chaque action est enfin caractérisée par son effet dans le temps, analysant si cet effet sur une intrusion est ***immédiat*** ou non et si la neutralisation de l'intrusion est ***ponctuelle*** ou non.

4 Etats de vulnérabilité d'un système

Nous allons définir un diagramme d'***états de vulnérabilité*** pour un site/une machine/un processus (appelé système) dans un contexte de sécurité [7], en relation avec la détection d'intrusion. Les transitions du diagramme d'état dépendent du ***rôle*** de l'action déclenchée.

- Dans l'état initial, on suppose le système dans l'état ***sain***, ce qui veut dire sans vulnérabilité connue et offrant la totalité du service.
- A la réception d'un diagnostic d'intrusion, le système passe dans l'état ***vulnérable*** où une vulnérabilité a été détectée et cette vulnérabilité est couramment exploitée. Il n'y a pas de dégradation de service.
- Une correction étant une action qui supprime la vulnérabilité en ne provoquant aucune perte de service, une correction réussie fait repasser le système dans l'état ***sain***.
- Si aucune correction n'est possible ou n'a fonctionné, l'ensemble des actions de compensation est divisé en deux catégories qui conduisent le système dans deux états différents : si la compensation induit une perte de service, on passe dans un état ***service dégradé*** non vulnérable, mais avec dégradation du service offert ; sinon la ***compensation*** (dite ***ponctuelle***), bien que neutralisant l'intrusion, fait passer le système dans un état ***vulnérable compensable*** sans perte de service.
- Si une action de compensation (au déclenchement automatique) a amené le système dans l'état ***vulnérable compensable***, une action de correction déclenchée manuellement fait repasser le système dans l'état ***sain***.
- Lorsque le système se trouve dans l'état ***service dégradé***, la fonction de réaction ne suffit pas pour revenir dans l'état ***sain***, il faut en effet que l'opérateur intervienne afin que le système offre de nouveau le service initialement prévu.

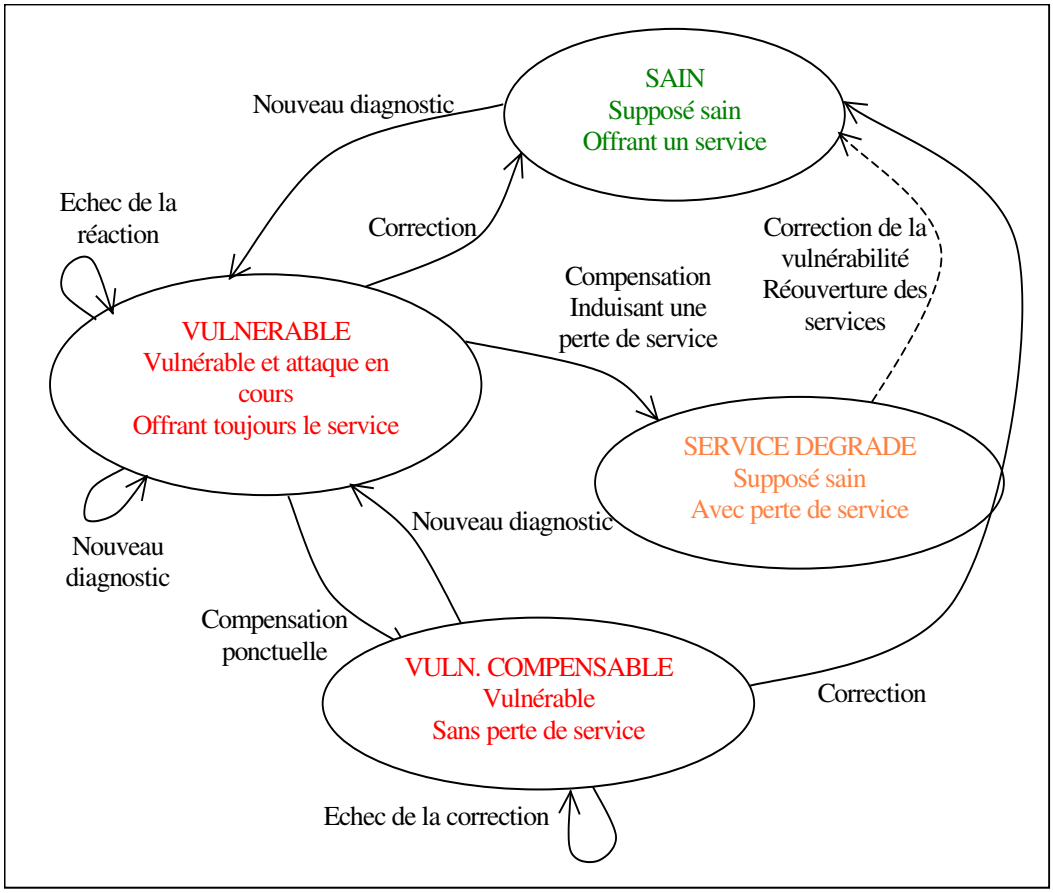


Figure 2. Diagramme d'états de vulnérabilité d'un système

5 Algorithme d'exécution des actions

La réaction fait suite à la détection d'une intrusion et est fonction de cette intrusion. Le **diagnostic** d'une intrusion précède la **décision** de lancer ou non une **réaction** constitué d'une ou plusieurs actions. A l'issue du lancement des actions le système attaqué se trouve dans un des états décrit dans la figure 2.

```
Etat du système = actions ( décision (diagnostic))
```

Figure 3. Interdépendance des paramètres de la fonction de réaction

Diagnostic

Le ***diagnostic*** d'une intrusion est l'étape qui précède la décision de déclencher ou non des actions. La qualité de ce diagnostic est essentielle du point de vue de la confiance que l'opérateur aura dans l'IDS.

Décision

La ***décision*** est la fonction qui choisit, en fonction de paramètres incluant le diagnostic d'intrusion, de lancer ou non une action donnée face à une attaque. Elle n'a pas les moyens (et ce n'est pas son rôle) de remettre en cause la qualité des diagnostics fournis par les autres éléments de l'IDS.

Algorithme

Tenant compte des diverses définitions et modèles présentés dans les paragraphes précédents, nous proposons l'algorithme suivant :

1. Exécution des actions au déclenchement ***automatique***, d'information, de dissuasion et de correction**.**
2. Si la correction a échoué, exécution des actions automatiques de compensation.
3. Proposition des actions au déclenchement ***manuel*** et armement d'un temporisateur.
4. Si l'opérateur ne réagit pas au bout du temps imparti, on lui signale par un message qu'il n'a pas réagi manuellement à cette attaque, aller en 7.
5. Sinon, exécution des actions manuelles d'information, de dissuasion, et de correction choisies par l'opérateur
6. Si la correction manuelle a échoué, on exécute l'action de compensation manuelle choisie.
7. Fin.

Figure 4. Algorithme de déclenchement des actions

Stratégie

Enfin, en nous inspirant des travaux de l'IETF sur les politiques de sécurité [3], nous avons introduit le terme de ***stratégie*** d'exécution pour chaque ensemble d'actions dont le déclenchement est ***automatisé***.

- Si la stratégie est ***Do until success,*** la série d'actions est exécutée jusqu'à ce que l'une des actions réussisse.
- Si la stratégie est ***Do All,*** la série d'actions est exécutée entièrement, quel que soit le statut d'exécution de chaque action.
- Enfin, si la stratégie est ***Do until failure*** , la série d'actions est exécutée jusqu'à ce que l'une des actions échoue.

L'exécution des actions automatiques dépend de la stratégie d'exécution choisie pour chaque groupe d'actions au niveau du fichier de configuration :

La fonction de réaction donne le nouvel état du système en résultat.

6 Exemples d'actions possibles dans des situations plausibles

Nous présentons ici des exemples d'action illustrant les diférents rôle que peut avoir une action. Dans ces exemples, les paramètres *nom* (classification name), *srcaddr*, *targetaddr*, *srcport*, *targetport*, *targetProcessPID*, etc, sont fournis par le diagnostic de l'intrusion. Nous supposons que nous pouvons de

plus lancer tout type de commande sur toute machine du réseau (local) comme c'est le cas dans la fonction réaction de Mirador présentée au paragraphe 7. Nous utilisons comme vecteur réseau, les fonctionnalités de NePoSe [6] en terme de coupure de communication (neposeTCPreset), pour interrompre une communication TCP en temps réel.

6.1 Information et Dissuasion

Information : Génération d'alarme

Il s'agit de messages à destination de l'opérateur de sécurité via l'interface de supervision de l'IDS. Cette action peut-être lancée après chaque détection d'intrusion.

- Exemple de commande :

  `logger -plocal7.alert "nom srcaddr srcport -> targetaddr targetport"`
- Système sur lequel est lancé la commande: syslog fonctionnant en réseau, la commande peut être lancée sur la machine hébergeant la fonction de réaction (ou sur n'importe quel autre système du réseau local).

Dissuasion : **Envoi d'un message vers le client (niveau applicatif)**

Sur diagnostic d'une intrusion concernant un service pour lequel on sait afficher un message sur le client, on génère le trafic correspondant. Ceci est réalisé par exemple par la fonction Block de eTrust (Annexe D) qui envoie un message configurable sur le client navigateur http.

- Pas d'exemple de commande connue disponible (cette fonction n'est pas pilotable à distance sur eTrust)

6.2 Correction

Suppression d'un service ne faisant pas partie du service offert par le site

Ceci peut consister à ajouter un filtre sur le garde-barrière, le routeur, ou le serveur, ou bien à détruire un processus, ou encore arrêter un démon. Par exemple, si un utilisateur a installé un serveur web personnel sur son poste, NePoSe le détecte comme contraire à la politique de sécurité et on peut empêcher les accès au démon *httpd* sur cette machine.

- Exemple de commande : interdiction des connexions http venant de l'extérieur sur un serveur personnel

  `ipchains -A input -i eth0 -s 0/0 -p tcp -d targetaddr/32 80 -l -j DENY`
- Système : Le firewall du réseau local.

6.3 Compensation

Coupure de communication

L'outil NePoSe [6] permet de couper une communication TCP, dont il a connaissance du contexte, via la commande `neposeTCPreset`.

- Exemple de commande :

  `neposeTCPreset srcaddr srcport targetaddr targetport protocol`
- Système : machine hébergeant le programme NePoSe.

Eteindre la cible/le firewall

Dans l'urgence, si les autres actions de compensation ont échoué, il pourra s'avérer utile d'éteindre une ou des machines pour stopper une intrusion, le temps que la vulnérabilité utilisée soit corrigée.

- Exemple de commande :

  `Halt`
- Système : Le firewall pour couper toutes les communications entre le site et l'extérieur, ou bien *targetAddr*.

Arrêter un service

Un processus, correspondant au service rendu par le site, et possédant une vulnérabilité, peut être détruit. Dans le cas des ports utiles au service offert par le site, la fermeture de port est une compensation, puisqu'il y a dégradation du service offert.

- Exemple de commande :
  `kill -9 targetProcessPID`
- Système : la machine sur laquelle est lancé le processus à détruire (*targetAddr*).
- Exemple de commande : fermeture d'un service web
  `ipchains -A input -i eth0 -s 0/0 -d targetaddr 80 -l -j DENY`
- Système : Le firewall du réseau local.

7 Fonction de réaction au sein du projet MIRADOR

7.1 Intégration à l'architecture

La fonction de réaction de Mirador s'intègre à l'architecture du projet (figure **5**). Elle traite les diagnostics transmis sous forme de message IDMEF [2] par le BEM [6]. A partir de la configuration, la fonction de réaction réalise plusieurs abonnements, lui permettant de ne recevoir que les diagnostics pour lesquels elle sait réagir. Après chaque réception de message, on appelle les fonctions de décision puis de déclenchement d'actions en utilisant les informations de configuration et des informations contextuelles véhiculées par le diagnostic.

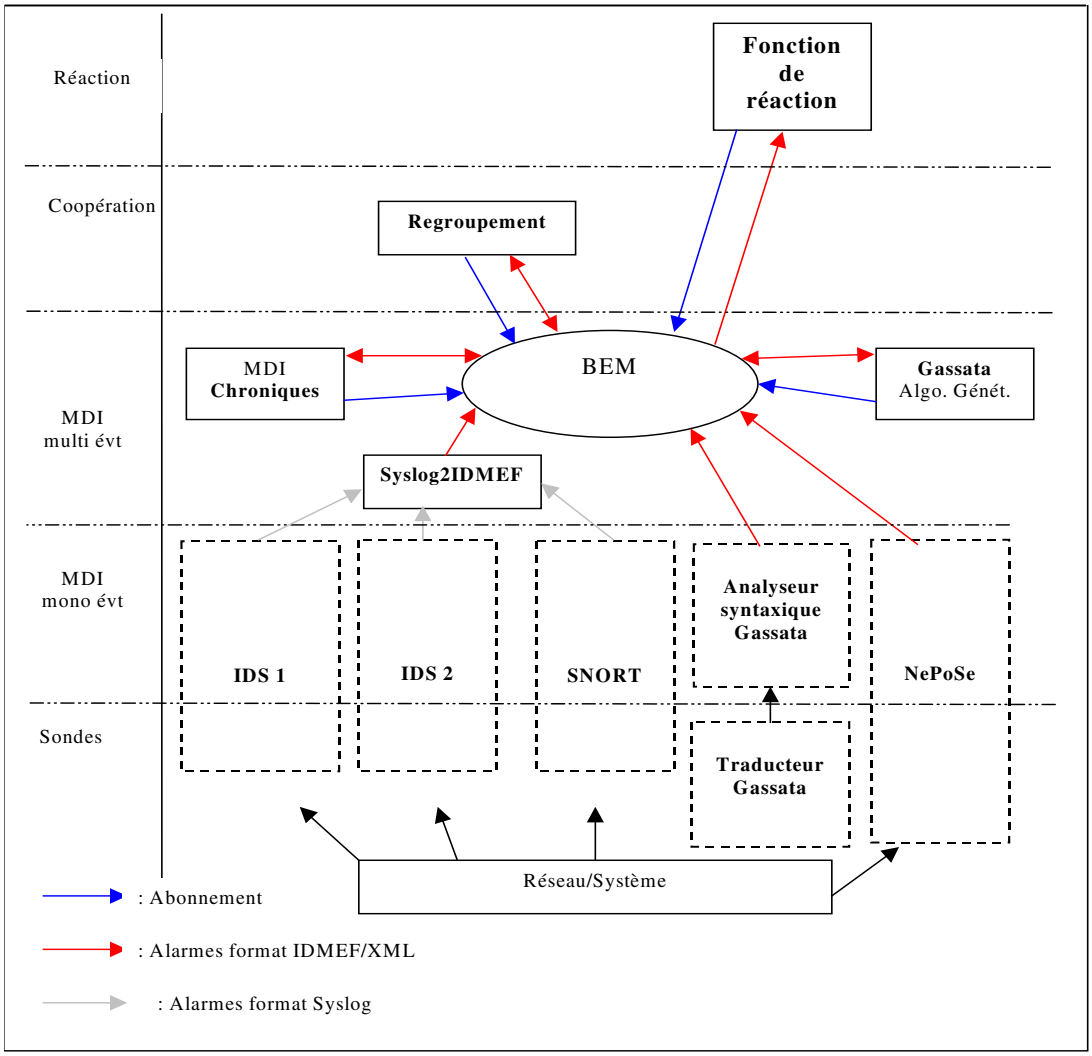


Figure 5. Collecte de diagnostics pour la fonction de réaction

7.2 Implémentation

Nous allons décrire les points importants de l'implémentation en commençant par le procédé de déclenchement des commandes à distance qui la clé de la flexibilité du mécanisme. Nous analyserons ensuite le fichier de configuration qui offre à l'opérateur un riche paramétrage adapté à l'algoritme présenté au paragraphe 4 ainsi que l'interface graphique.

Exécution de commande à distance

Pour une ouverture maximum, nous souhaitions pouvoir lancer n'importe quelle commande sur n'importe quelle machine de manière sécurisée. L'utilisation de *ssf* (shell sécurisé français [5]) nous offre cette souplesse, gage de pérennité du mécanisme de réaction proposé (figure 6).

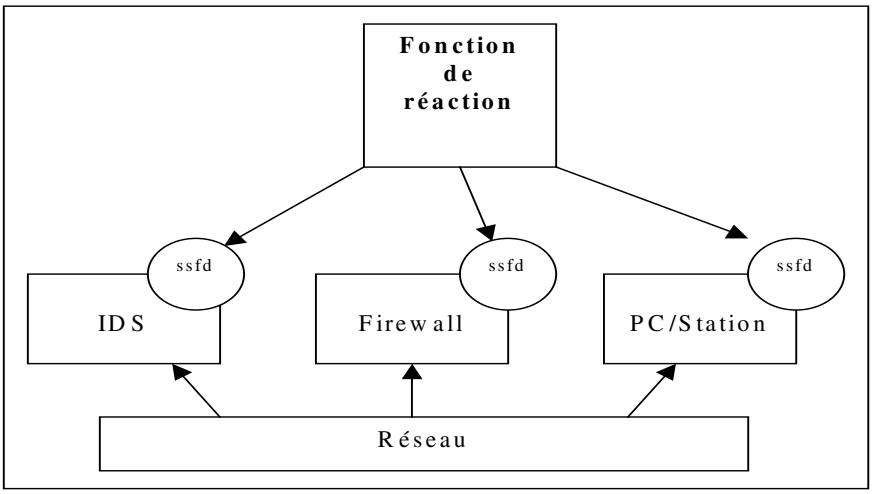


Figure 6. Lancement des contre-mesures par la fonction de réaction de MIRADOR

Fichier de configuration

Grâce au fichier de configuration, l'opérateur a la possibilité de configurer la fonction de réaction de manière à profiter au mieux des possibilités modélisées dans cet article. Le fichier est au format XML et un exemple en est donné en annexe. Dans ce fichier, l'opérateur définit autant d'éléments *Alarme* que de classes d'attaques auxquelles il veut réagir. Pour chaque élément *Alarme*, l'opérateur doit définir un attribut *id* et un élément associé *Abonnement* uniques.

L'élément *Abonnement* reprend les exigences de la DTD IDMEF[3] pour l'élément *Alert*, mais avec les éléments et attributs optionnels ; cet élément permet de s'abonner à différents types de messages avec la flexibilité maximale offerte par le BEM.

L'attribut *id* identifie l'abonnement. On compare seulement l'attribut *id* avec la valeur transmise par le BEM lors de la réception d'un message IDMEF.

L'opérateur décrit ensuite, dans les éléments correspondant chacun des ***rôles*** possibles, les commandes à exécuter en réaction aux attaques correspondant à cet élément *Alarme*. Il peut préciser grâce à l'attribut *strategie* une ***stratégie*** d'exécution prise parmi *all*, *until failure* ou *until success*, la valeur par défaut de cet attribut étant *all*.

Pour chaque commande, l'opérateur peut ensuite indiquer si son déclenchement doit être automatisé ou s'il doit valider le choix (attribut *declenchement* égal à *manu*, valeur par défaut : *auto*) et il doit donner l'adresse IP d'exécution ou le champ correspondant dans le message IDMEF en indiquant *Sonde*, *Cible* ou *Source*. Le cas échéant, il décrit les paramètres (et/ou options) de chaque commande de la même façon (directement ou en désignant un champ du message IDMEF par l'intermédiaire d'une notation pointée).

Interface Graphique

L'interface graphique permet à l'opérateur de visualiser les messages IDMEF reçus du BEM, de suivre l'exécution des commandes lancées automatiquement, de choisir des actions manuelles à exécuter et enfin de lire les messages d'erreur et de contrôle produits par la fonction de réaction.

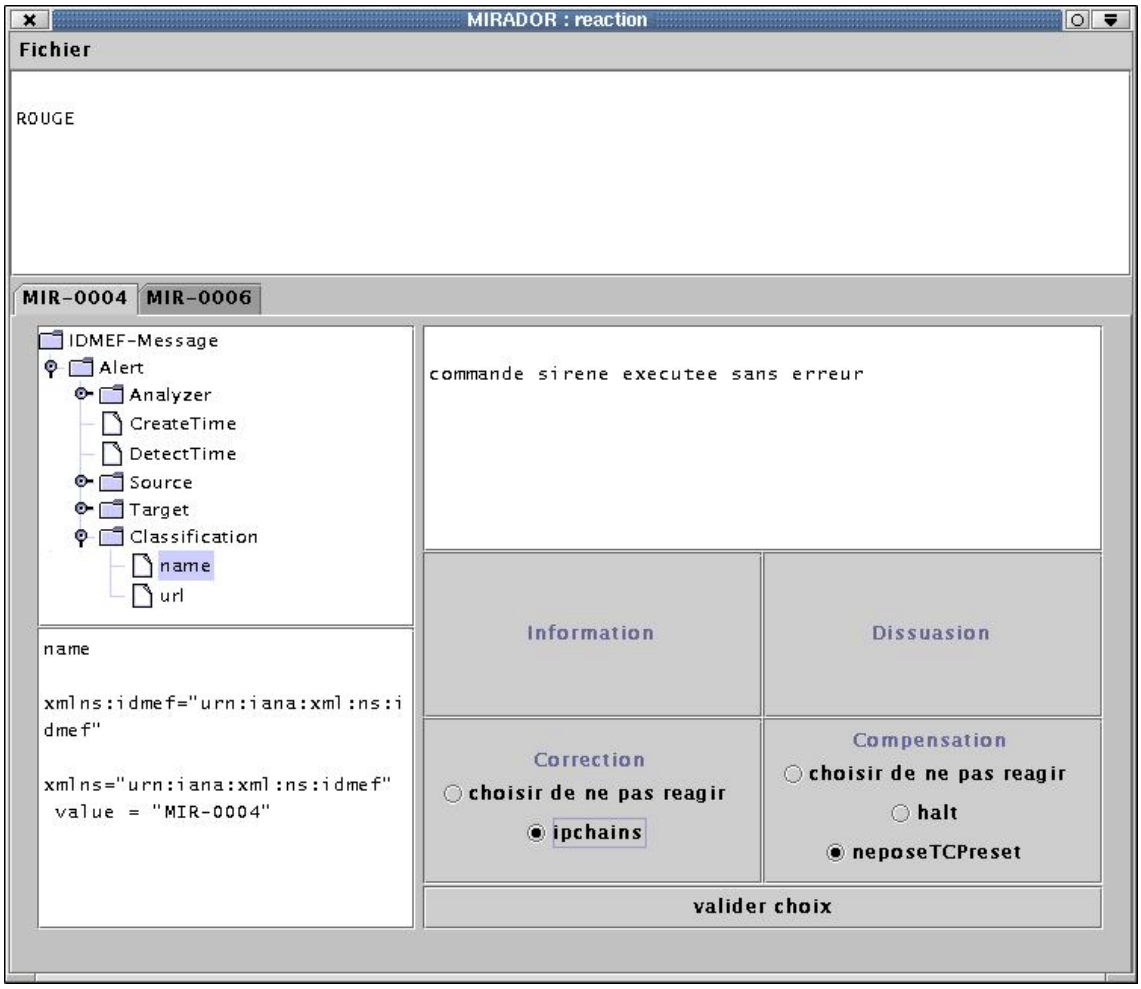


Figure 7. Interface Graphique

8 Conclusion

Dans ce document, nous avons décrit la fonction de réaction réalisée dans le cadre du PEA MIRADOR. A ce jour, une version expérimentale développée en Java est opérationnelle. Son originalité réside dans l'algorithme proposé au paragraphe 4 et dans l'utilisation faite des éléments de la taxonomie des actions décrite au paragraphe 2. Son intérêt est aussi dans le nombre infini d'actions possibles : toute commande exécutable via ssf est une action potentielle donnant toute sa flexibilité au procédé. Nous avons montré plusieurs exemples plausibles d'utilisation de cette fonction, toutes expérimentées dans le cadre de MIRADOR.

Il faut maintenant affiner une politique de réaction élaborée en construisant un fichier de configuration plus complet, associant des catégories d'attaques à des ensembles de contre-mesures.

Ensuite, le modèle de décision pourra être amélioré, afin de ne pas rejeter en bloc les actions dès qu'il manque un paramètre pour l'une d'entre elles. Pour cette évolution, on pourra se baser sur les attributs *strategie* et *role* du fichier de configuration.

9 Remerciements

Ce travail été financé par le DGA/CELAR/CASSI dans le cadre du PEA MIRADOR. Les auteurs souhaitent également remercier l'ensemble des participants au projet MIRADOR : Jacques Capoulade, Samuel Dubus, Aldric Feuillebois (Alcatel CIT), Frédéric Cuppens, Alexandre Miège (CERT), Patrice Carle (ONERA), Ewan Cochevelou, Jean-François Lejard, Maryse Le Goff (ENST-Bretagne), Laurent Heye, Ludovic Mé et Cédric Michel (Supélec Rennes).

10 Références

[1] Etat de l'art sur les réactions – Document interne au projet Mirador
Maryse Le Goff, Sylvain Gombault.

[2] I9ntrusion Detection Message Exchange Format Data Model and XML Document Type Definition
draft-ietf-idwg-idmef-xml-03.txt
Hervé Debar, David A. Curry.

[3] Policy Core Information Model Extensions
draft-ietf-policy-pcim-ext-01.txt
B. Moore, L. Rafalow, IBM, Y. Ramberg, Y. Snir, J. Strassner, A. Westerinen, Cisco Systems, R. Chada, Telcordia Technologies, M. Brunner, NEC, R. Cohen, Ntear LLC

[4] BEM : Bus d'Evénements Mirador – Document interne au projet MIRADOR
Alcatel - ENST Bretagne

[5] SSF : http://ccweb.in2p3.fr/secur/ssf

[6] NePoSe : Network Policy Sensor - Document interne au projet Mirador
Sylvain Gombault, Ewan Cochevelou

[7] Guide de la sûreté de fonctionnement
J. -C. Laprie, Cépaduès - Editions 1995

Annexe : Exemple de fichier de configuration

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE Config SYSTEM "Conf.dtd">
<Config>
 <Alarme id="MIR-0004" >
  <Abonnement> <Alert><Classification><name>MIR-0004</name></Classification></Alert>
  </Abonnement>
   <Information strategie="untilSuccess">
      <Commande nom="logger" IPaddr="sonde">
         <Param>#-plocal7.alert " </Param>
         <Param>IDMEF-Message.Alert.Classification.name</Param>
         <Param>IDMEF-Message.Alert.Source.Address.address</Param>
         <Param>IDMEF-Message.Alert.Source.Service.port</Param>
         <Param>#-></Param>
         <Param>IDMEF-Message.Alert.Target.Address.address </Param>
         <Param>IDMEF-Message.Alert.Target.Service.port </Param>
         <Param>#"</Param>
      </Commande>
      <Commande nom="sirene" IPaddr="firewall">
      </Commande>
   </Information>
   <Dissuasion strategie="untilFailure">
      <Commande nom="com3" declenchement="manu" IPaddr="cible">
      </Commande>
      <Commande nom="com4" IPaddr="192.168.56.102">
      </Commande>
      <Commande nom="com5"  IPaddr="source">
         <Param> #192.168.56.97</Param>
      </Commande>
   </Dissuasion>
 </Alarme>
 <Alarme id="web" >
   <Abonnement> <Alert><Target><Service><port>80</port></Service></Target></Alert>
   </Abonnement>
   <Correction strategie="all">
      <Commande nom="com6" declenchement="manu" IPaddr="sonde">
      </Commande>
      <Commande nom="com7" IPaddr="firewall">
      </Commande>
      <Commande nom="com8"  IPaddr="firewall">
      </Commande
   </Correction>
   <Compensation>
      <Commande nom="neposeTCPreset" declenchement="auto" IPaddr="Nepose">
         <Param>IDMEF-Message.Alert.Source.Address.address</Param>
         <Param>IDMEF-Message.Alert.Source.Service.port</Param>
         <Param>IDMEF-Message.Alert.Target.Address.address </Param>
         <Param>IDMEF-Message.Alert.Target.Service.port </Param>
         <Param>IDMEF-Message.Alert.Source.Service.protocol</Param>
      </Commande>
      <Commande nom="halt" declenchement="manu" IPaddr="firewall">
      </Commande>
   </Compensation>
 </Alarme>
</Config>
```

Page intentionnellement blanche

This page has been deliberately left blank

Click here to view slide presentation presented in PDF format

High-Efficient Intrusion Detection Infrastructure

Thomas Holz, Michael Meier, Hartmut Koenig
Brandenburg University of Technology Cottbus
Department of Computer Science
PF 10 13 44, 03013 Cottbus
Germany

Tel./Fax: +49-355-69-2236/2127
email:{thh,mm,koenig}@informatik.tu-cottbus.de

Abstract

In recent years research activities in computer network security focus more actively on the development of effective methods in intrusion detection. The reason for this development is the rapidly increasing potential of threats to social, economical, and military information stored in information technology (IT) systems. Powerful and practically applicable mechanisms are required to protect critical infrastructures. Intrusion detection systems have been proven as a powerful means for the detection of IT security violations. They provide protection of computer and network resources by automatic detection of security violations. Some of these systems are able to initiate appropriate intrusion response actions. The crucial point for real-time applications, especially for host-based audit analysis, is the detection speed. In the paper we present the distributed intrusion detection infrastructure HEIDI which tackles this problem. HEIDI provides a module system based on sensors and agents to set up tailored intrusion detection systems for real-time applications. The basic features of the HEIDI approach are a distributed analysis functionality, the handling of overload situations, and a dynamic configurability. Furthermore, the problem of time-consuming audit analysis is compensated by integration of StraFER, a new signature match algorithm.

1 Motivation

The rapid advance of communication technologies in many areas of the human society accelerates the shift of many social processes on such systems, in particular on the Internet. This brings numerous benefits to the users, but it also increases their dependencies on these technologies. These dependencies as well as the technological complexity of the systems themselves create an increasing potential of threats for these systems which make them more and more vulnerable. The constantly growing number of computers in the Internet gives hackers, crackers, terrorists, and subversive insiders better and better opportunities for attacks. This concerns not only the critical infrastructures of industrial states, the trade, the information system, or the health care system, but in particular the military protection of the society [1, 2].

To counter these threats besides preventive measures such as authentication of communication partners, encryption of communication, and access control to resources, means on the technological level are required which allow to detect and indicate security violations to evaluate and to possibly confine the damage.

Intrusion detection systems (IDS) have proved to be an effective instrument for this purpose. Systems with real-time capabilities provide automated protection of computer and network resources and allow the detection of ongoing security violations. Intrusion detection systems are currently one of the few reactive security mechanisms to counter threats on the communication infrastructure. They have been developed since 20 years and successfully deployed in practice [3]. Various commercial solutions are available. The effectiveness of commercial systems in real-life environments, however, is limited. They mainly confine themselves to detecting simply structured security violations in a post-mortem mode. For the deployment in large computer networks, they are less suited, especially for tight time constraints. Growing communication infrastructures and increasing user requirements raise new problems (e.g.

Paper presented at the RTO IST Symposium on "Real Time Intrusion Detection", held in Estoril, Portugal from 27-28 May 2002, and published in RTO-MP-101.

encryption, switching technologies) and demands (e.g. privacy) which are not covered by existing concepts.

In the paper we present the distributed intrusion detection infrastructure HEIDI that is currently being developed at Department of Computer Science at Brandenburg University of Technology Cottbus. It is based on the experience we gathered with the intrusion detection system AID [4] developed in our group and other systems as well. The objective of the HEIDI approach is to get a module system to flexibly set up intrusion detection systems for real-time applications. The HEIDI concept is based on the use of sensors and agents which can be combined to set up a tailored intrusion detection system which meets the requirements of a given application environment. Unlike other approaches, which try to reduce the amount of audit data, the HEIDI concept aims at the complete evaluation of all audit data. This is achieved by using a combined analysis distribution, the delegation of processing functionality in overload situations and the application of a new optimized signature match algorithm. The paper describes the basic principles of the HEIDI approach. Section 2 formulates requirements to the design of modern intrusion detection systems, which inspired the HEIDI development. Section 3 introduces the basic features of the HEIDI approach. It gives examples for the surveillance of different IT systems with HEIDI-based architectures. In Section 4 we describe the new match algorithm StraFER which is applied in HEIDI agents. The conclusion summarizes the achievements of the approach and gives an outlook on next research steps.

2 Requirements to modern intrusion detection systems

The security function intrusion detection deals with the monitoring of IT systems to detect security violations. Due to the large amount of incoming audit data this analysis can be only efficiently processed if automated evaluation support is given. The decision which activities can be considered as security violations in a given context is determined by the applied security policy. For the detection of security violations, mainly two complementary approaches are applied: anomaly detection and misuse detection. Anomaly detection aims at the exposure of abnormal user behavior. Misuse detection focuses on the automatic detection of known attacks. These attacks are described by patterns, so called signatures, required to identify an attack in an audit data stream. Misuse detection has revealed to be the more effective approach. It is only considered in the sequel.

Intrusion detection systems are usually applied to monitor critical servers and complete (local) computer networks. The observation of desktop computers represents more an exception. For the observation, stand-alone and distributed intrusion detection systems can be distinguished. Intrusion detection systems are often deployed in connection with other security mechanisms like firewalls to support a superior security management. Therefore distributed solutions are more and more becoming a typical characteristic of modern intrusion detection systems. The deployment of intrusion detection systems in practice has revealed a couple of problems such as the high amount of audit data, privacy, response mechanisms, insufficient system protection, high false alarm rates and others for which different solutions have been proposed. In the discussion here we focus on the following problems: detection efficiency, and system adaptation and maintenance. We consider in particular

1. the use of distributively acting IDS components,
2. the efficient development of signatures, and
3. the integration of efficient analysis methods.

In this context, our main interest aims at the minimization of the time intervals between the appearance of security violations and their detection to improve the chances of effective intrusion response actions. Unlike network based intrusion detection systems, host based systems often provide no chance to stop an ongoing attack, because audit records are mostly only reflections of completed security relevant actions. In case of relatively long attacks, however, a fast audit analysis may be able to reduce the arising damage.

2.1 Distributively acting IDS components

Modern distributed intrusion detection systems consist of a set of modules for capturing, preprocessing, analyzing the audit data, and for archiving them if required. In addition, most commercial products offer an appropriate management functionality. As far as efficiency is concerned, the primary aspect of such a

system is the distribution of time-consuming analysis functionality. Two categories of intrusion detection systems can be distinguished: systems with a centralized and a decentralized analysis. Systems with a centralized analysis are simpler to implement, but the analysis function represents a performance bottleneck and a single point of failure. Further large amounts of data have to be transferred between a local host and the central processing unit. These effects we could also observe while testing our own centralized system AID (*Adaptive Intrusion Detection system*) [4]. If several processing units are used synchronization is required, but load distribution and efficiency are essentially better. Furthermore the processing units can be aligned in a hierarchical manner, e.g. in a two-layered scheme, where the units at the lower level perform a detection of local attacks and a unit at the upper level is responsible for finding all distributed security violations.

Most approaches, however, apply the centralized processing paradigm [5]. Only a few systems use the distributed approach as, for instance, DIDS, CSM, AAFID, and EMERALD. In this context DIDS (*Distributed Intrusion Detection System*) was the first intrusion detection system that combined local audit evaluations, network monitoring and a central alarm correlation [6]. CSM (*Cooperating Security Managers*) implemented an unusual idea [7]. Here every monitored host contains a security manager. When a user logs in first time on a certain host the security manager of this host is responsible for recording and analyzing all subsequent actions of this user, even if he moves to another host within the network. The roaming of users, however, cause an enormous data transfer. The approach is therefore not applicable to a fast inspection of large amounts of audit data. The AAFID (*Autonomous Agents for Intrusion Detection*) concept, on the other hand, addresses the problem of system load caused by using intrusion detection systems. The main idea here consists in the application of many small, specialized and hierarchically grouped entities [8]. These components (filters, agents, transceivers and monitors) are too limited in their performance to meet the requirements of an efficient audit analysis. The fourth systems, EMERALD (*Event Monitoring Enabling Responses to Anomalous Live Disturbances*), is a military sponsored development which aims at the application of a flexible set of complex modules, the EMERALD monitors. These monitors integrate both detection and response functionality, and are designed to be interoperable with many other security functions at a very high degree [9]. They can be connected among each other within a three-layered hierarchy. An EMERALD intrusion detection system is able to protect large networks, especially in critical enterprise environments. This approach, as well as others, does not aim at a high audit analysis speed. There are no documented performance data available for these systems so far.

Further problems that are related to the design of distributed intrusion detection systems are a high runtime adaptability, robustness, availability and fault tolerance. In particular in sensitive environments there is a fundamental operational need for the survivability of the intrusion detection systems under various conditions. Every time interval, in which the system is not running, represents a threatening situation. Modern distributed intrusion detection architectures used for the protection of critical infrastructures should ensure this dynamic adaptability as extensively as possible.

2.2 Efficient development of signatures

The goal of misuse detection systems is to automatically find traces of known attacks in audit data streams. The signatures have to be expressed in a representation that can be used as basis for the analysis process. The derivation of signatures based on a given security policy has often proved to be a crucial point in audit-based misuse detection. The problem can be divided in two steps:

1. to determine attack patterns to be searched for during the analysis, and
2. to encode the patterns in an appropriate form.

The determination of patterns that are significant for an attack usually involves an in-depth investigation of the constituent actions of known attacks. This step requires a lot of expertise about attacks and audit functions. The identified signatures have to be expressed in an adequate representation that can be used as basis for the analysis process. Since the representation of signatures is specific to the used detection mechanisms most intrusion detection systems use their own language to encode signatures. Many existing intrusion detection systems (e.g. EMERALD, ASAX [10], AID) use rule-based languages to encode signatures. Others use state transition diagrams (STAT [11]) or petri net representations (IDIOT [12]).

Although these approaches have shown their usefulness, they lack some important features. Most intrusion detection systems require not only a description of the signatures but also details on the manner the detection process has to work. Often adequate means to build abstraction in developing signatures are missed. This makes the development and maintenance of signatures complicated and error-prone.

2.3 Integration of efficient analysis methods

The analysis of the audit data is still the bottleneck for intrusion detection systems, especially for real-time applications. Although analysis techniques have improved continuously, the capability of knowledge-based methods, which are usually applied in intrusion detection systems, remains limited. The increasing amount of audit data often wipes out the progress. Audit bursts like, for instance, 100 Mbps for usual PCs running Microsoft Windows 2000 requires special optimized algorithms.

Concerning the analysis techniques misuse detection systems can be roughly divided into two groups. The first group of systems uses the specified signatures to generate program code (e.g. in C/C++) that searches for attack patterns using the specified criteria. Systems of the second group use the expert system approach to analyze audit data.

Examples for intrusion detection systems of the first group are IDIOT and the STAT tool suite. These systems create a program module for each signature that after its compilation can be used to search for the specified patterns in audit data. A performance issue here is that different signatures are checked or executed independently. Thus it is scarcely possible to take advantage of the optimization potential that can be used if signatures are analyzed together. For example, techniques like condition sharing to avoid redundant calculations of conditions in a signature set cannot be used.

AID, ASAX, and EMERALD's P-BEST component [13] are examples for systems of the second group. Here the inference mechanism and match algorithm of the underlying expert system shell determine the used analysis technique. The most popular and applied match algorithm of these systems is RETE [14] (others are TREAT [15] and LEAPS [16]). The issue is that match algorithms of expert system shells are general-purpose algorithms. It is well known that a match algorithm may outperform all other algorithms in a specific application domain [17]. Therefore it depends on the concrete application domain which algorithm performs best. To our knowledge it is an unanswered question which match algorithm is best suited for misuse detection expert systems. Further the question arises, whether a special-purpose match algorithm for misuse detection systems can be constructed that noticeably outperforms general-purpose algorithms by taking advantage of known information about the application domain.

3 The HEIDI approach

The objective of the HEIDI approach (*High-Efficient Intrusion Detection Infrastructure*) is to provide an infrastructure for setting up tailored intrusion detection systems to speed up the detection capability. The term "infrastructure" means that a module system is defined which can be adapted to a specific intrusion detection architecture for a given target environment and application scenario, respectively. The main characteristics of such an architecture are the placement of necessary HEIDI modules and the general specification of their interconnectivity. Further refinements of the architecture towards a real intrusion detection system can be introduced by the integration of target-specific adaptations, e.g. interfaces for capturing host-specific audit data.

3.1 Functional overview

HEIDI distinguishes 3 basic components: sensors, agents, and user interfaces. The sensors collect and preprocess audit data. The agents provide the analysis units. They can cooperate among each other. The user interfaces serve for system management and user interactions.

3.1.1 HEIDI SENSORS

HEIDI sensors are specialized modules to collect and to handle audit data. They aim at a fast reading, preprocessing and forwarding of the audit data. Sensors can be placed at different points of the monitored hosts depending of the applied security policy. Different sensors at one host are coordinated by the supervising local agent. Figure 1 depicts the generic structure of a sensor.

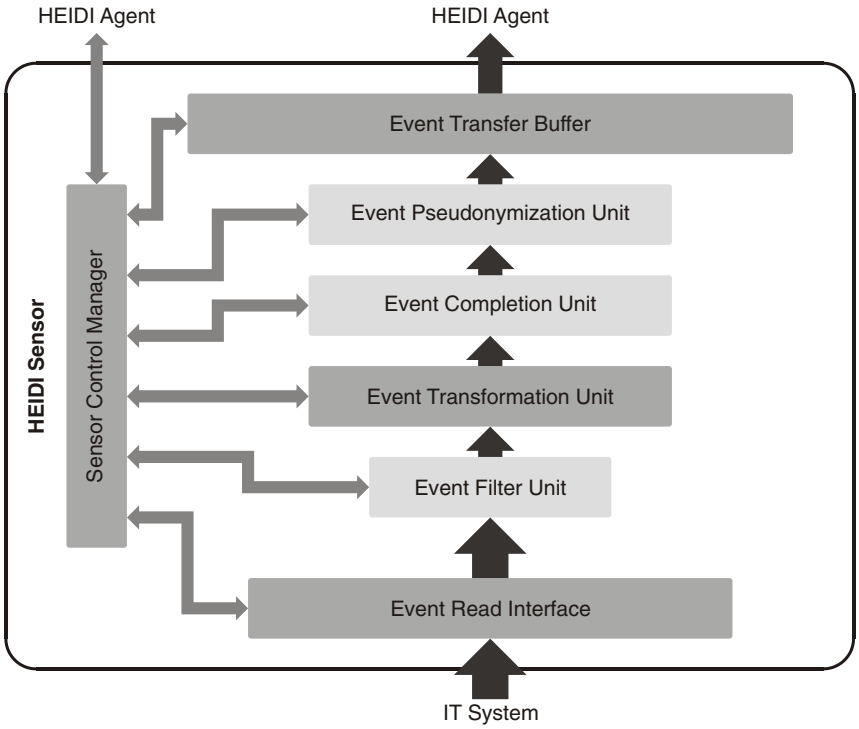


Figure 1: Structure of a HEIDI sensor

HEIDI sensors consist of permanent components (illustrated in darker grey), e.g. the read interface, the transformation unit for data converting, and the transfer buffer. Other components are optional (illustrated in lighter grey), e.g. the pseudonymization unit for encrypting user identifying references in the audit data like the user ID, the group ID and others to ensure user privacy. Most components of the sensor are connected by a data processing pipeline. They are supervised by the control and configuration component. After reading the information from the local host, the data are preprocessed and forwarded to the local HEIDI agent.

3.1.2 HEIDI AGENTS

After the fast capturing of the audit data by the sensors the second step to maximize the detection speed is to ensure an efficient analysis of these data. This is the task of the HEIDI agents. Beside the application of optimized analysis algorithms (see Section 4) they use an appropriate distribution of data. This distribution is based on a classification of the signatures into local and distributed contexts. To detect signatures with a local context only locally preprocessed data are analyzed, while for signatures with a distributed context data from various agents are demanded.

The most efficient way to perform such an analysis in a network is to apply a combined execution scheme. Similarly to systems like DIDS, AAFID, and EMERALD, HEIDI prefers to match signatures with local context on the corresponding host. The detection of distributed attacks takes place on a central location. Unlike any other known system, however, HEIDI applies this hybrid concept in a stringent manner to achieve a maximal local concentration and a minimal need for network traffic and delay. For this purpose we extended the notion of signature. In context of HEIDI signatures are not only used for mapping complete security violation sequences. A signature can also represent a partial sequence of such an attack. This extension enables a hierarchy of agents to split the detection process for a distributed attack into a number of local sub-detections and a small amount of central combining. This principle is not applicable to all distributed attacks, but some critical security violations, like several doorknob rattling variants, can be easily detected this way.

For the execution of all local and central detection processes, each involved host contains a single stationary HEIDI agent. Figure 2 shows the structure of an agent.

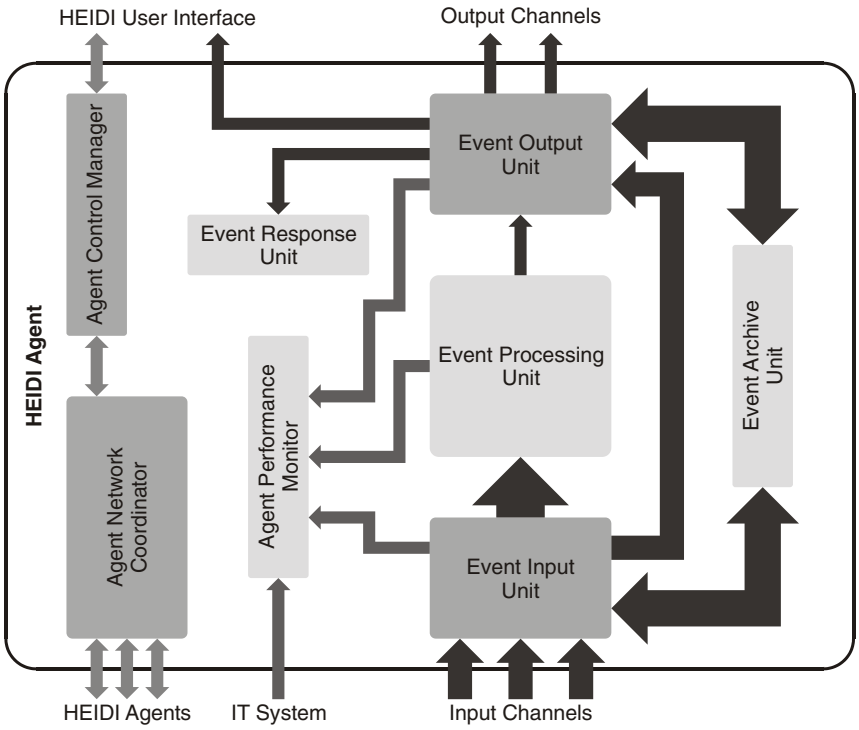


Figure 2: Structure of a HEIDI agent

A HEIDI agent receives the preprocessed information from all local sensors. It contains a central data processing pipeline which consists of an input, a processing, and an output unit. Input and output units are responsible for receiving, synchronizing and transmitting the security relevant data. The analysis of the audit data takes place in the processing unit. Several analysis processes can run in parallel, e.g. a complete signature detection, a fast signature detection for particularly critical attacks, and a simple audit statistics. The analysis algorithm applied currently is StraFER which is introduced in Section 4. In addition to the detection capabilities an agent can contain an active response unit which is able to initiate appropriate local countermeasures.

3.1.3 HEIDI USER INTERFACES

A HEIDI user interface is a graphical application which enables a security operator to perform several tasks in the context of a given HEIDI intrusion detection system. The most important tasks are the configuration of the system and the visualization of the detection results. Furthermore, a user interface can act as a link between a security management and a HEIDI intrusion detection system.

Every HEIDI agent provides a single interface for the connection with a user interface. This connection can be either local or remote. Thus a user interface can dynamically connect to a number of agents. Since several user interfaces can be attached to a HEIDI system at the same time every interface has to periodically read the corresponding configuration data.

3.1.4 HANDLING OF OVERLOAD SITUATIONS

For assuring a continuous, robust, and efficient intrusion detection operation, HEIDI uses an adaptive mechanism to compensate temporary overload situations. In conventional systems such overload situations, e.g. an audit burst, normally stop the execution of the intrusion detection system or cause a crash. To avoid this HEIDI agents are able to delegate analyzing functionality to other agents. A destination agent receives the preprocessed data and the analysis state. The delegation functionality, the required number of destination agents, and the duration of the delegation depend on several conditions. They are calculated and negotiated dynamically. Normally, a re-delegation is carried out when the overload situation has disappeared. To estimate the load situation in the intrusion detection system a HEIDI agent contains a monitor which evaluates the performance of both the host and some time-consuming agent components (see Figure 2).

3.2 Setting up intrusion detection architectures

HEIDI sensors and agents can be combined to set up a hierarchical intrusion detection architecture for a given target environment, e.g. a host or a local area network. Depending on the network structure and the applied security policy, special sensors and internal communication schemes can be configured. In this context, connectivity and data stream configuration have a special importance. For every security violation to be detected, it must be determined which subset of modules is involved and where the data analysis is appropriately located. To offer a flexible and efficient setup, an agent also can act as a transceiver, i.e. it does not analyze, or as a delegation server. The latter, which describes a HEIDI agent on demand, is required for enterprise networks with high failure safety requirements.

Figure 3 shows two different intrusion detection architectures. The left example shows a two-layered hierarchy, the right example a three-layered. All illustrated hosts (the greater rectangles) are equipped with two sensors (smaller embedded rectangles) and the corresponding agent (greater embedded rectangle). Streams of preprocessed audit and result information are illustrated as arrows, whereas the thickness of the arrows serves as an indicator for the transfer rates between the modules. In the left example all agents are working on detection processes, and the agent at the upper level is responsible for finding attacks with distributed context. In the right example, the white illustrated agents do not perform any analysis, so that their superior agent has to deal with a relatively high amount of incoming data. In this example, an agent at the third level serves as an overall result collector.

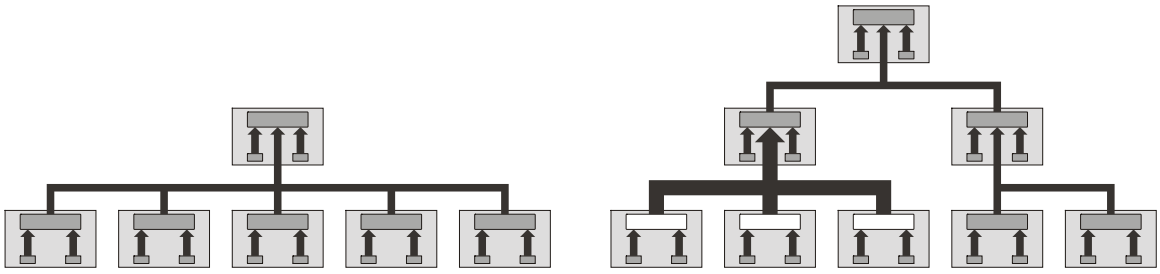


Figure 3: Examples of two HEIDI-based intrusion detection architectures

Figure 4 shows the expansion of the left architecture from Figure 3 by the integration of two delegation servers. The hosts, on which these delegation servers run, do not have sensors. In Figure 4 the upper depicted delegation server is configured to exclusively help the upper-level agent in the regular detection hierarchy. The lower server is dedicated to handle overload situations for all low-level hosts in the regular detection hierarchy. The left example shows a burst situation at two low-level hosts. The lower server overtakes in this case the analysis of the data. In the right example, there are also two hosts in an overload state. One of them is the upper-level host in the regular detection hierarchy. In this case the processing capacities of both delegation servers are used.

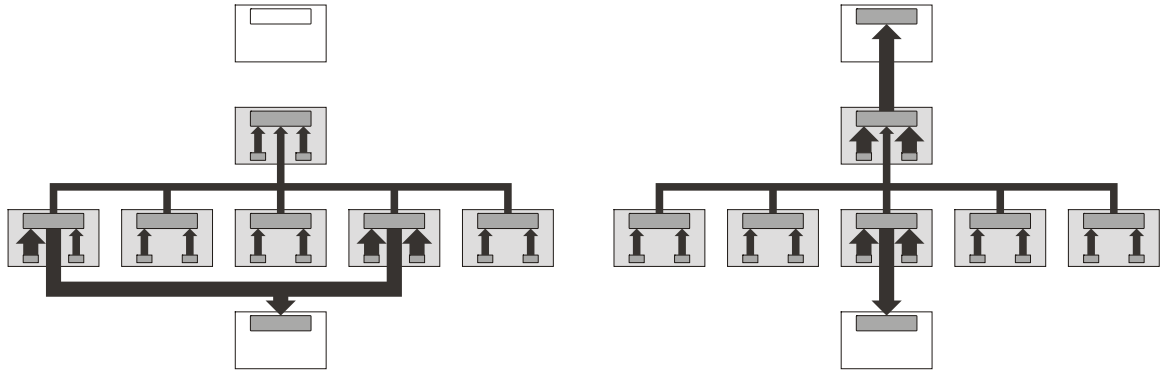


Figure 4: Scenario with two different overload situations for a single architecture

4 Efficient detection components

Beside the distributed collection and processing of audit data HEIDI provides means for an efficient analysis. Unlike many other systems HEIDI agents do not use a rule-based analysis. Instead the IDS specific analysis algorithm StraFER has been developed. This algorithm takes advantage of knowledge about the static structure of attack signatures to reduce the processing time. StraFER is based on the event description language SHEDEL which has been defined to facilitate the description and introduction of signatures into agents. In the sequel we first shortly introduce SHEDEL. Thereafter we give an overview of the StraFER algorithm.

4.1 The attack description language SHEDEL

To simplify the development and maintenance of signatures we developed the attack description language SHEDEL (*Simple Hierarchical Event Description Language*). The main objective of SHEDEL is to provide means to describe signatures without determining the detection process.

The main abstraction in SHEDEL is the event. An event can represent an audit record, an attack signature, a Meta attack signature and so on. Describing a signature in SHEDEL means to specify an event. An event consists of a collection of one ore more sub-events, also called steps, which are related to each other, temporally or by their properties. There is a set of basic events which cannot be divided in sub-events. They represent the basic recognizable units, e.g. audit records or network packets. Thus it is possible to specify a hierarchy of events that can be used to describe a pattern of audit records. To support the response to an attack (or any event) it is possible to link an action to sub-events of an event specification.

An event is characterized by a name and a set of properties called features. The name is used as reference for other events. The features contain relevant information about an event. A basic event contains all information of the corresponding audit record, e.g. the names of involved users and objects are represented as features of the event. For non-basic events, the features must be defined. A feature can be defined by referring to a feature of a step. Thus a feature definition is a pair consisting of a name and a feature of a step contained by this event.

For the successful completion of an attack, its constituent actions usually must be executed in a specific order. To describe the signature of such an attack in SHEDEL it must be possible to specify the order of steps of an event. This can be done by defining a predecessor of a step. Additionally to the correct order of actions typically further conditions must be fulfilled to complete an attack successfully. An example for such a condition is that the same user must execute all actions of an attack. To describe signatures for such kind of attacks SHEDEL allows the definition of conditions within an event.

Figure 5 shows the description of an event `E` that consists of the steps `A` and `B`. The features `start` and `filename` of `E` are defined using the features `time` and `file` of step `A`. The `username` feature of `E` is mapped to the `user` feature of step `B`. The right side of Figure 5 shows the resulting event hierarchy.

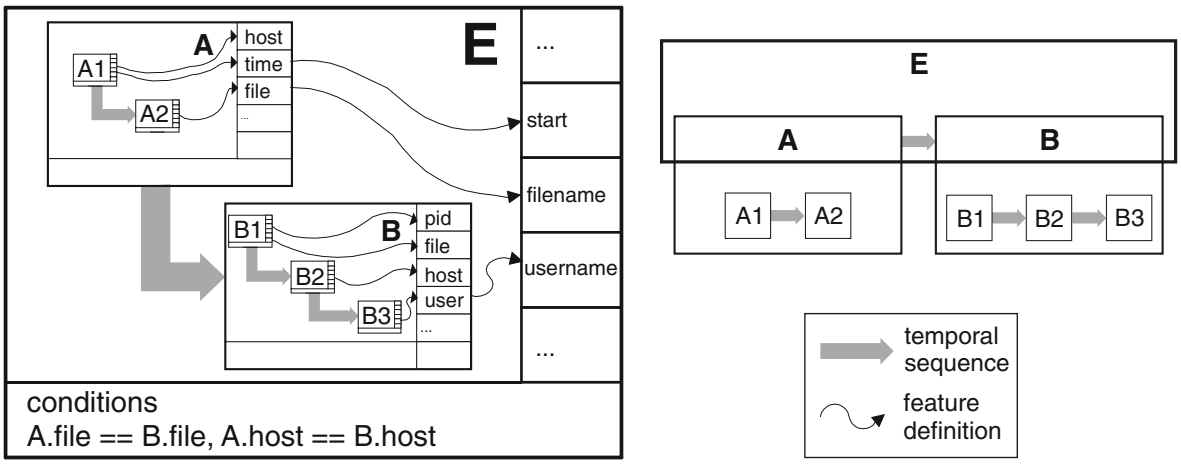


Figure 5: Definition of an event and the resulting event hierarchy

The description of the illustrated event `E` in SHEDEL looks as follows:

```
EVENT E
{
  STEP step1
    INITIAL                  // step1 is the initial step
    TYPE A
  STEP step2
    EXIT                     // step2 is the final (completing) step
    TYPE B
    REQUIRES step1           // step1 is predecessor of step2
  ACTIONS
    ...                      // any actions
  CONDTIONS
    step1.file == step2.file,
    step1.host == step2.host
  FEATURES
    start = step1.time,
    filename = step1.file,
    username = step2.user
}
```

Note that no operational details about the detection process are contained in this description. Using hierarchical event description it is possible to define adequate abstractions to reason about intrusions. So it is simple to define a generalization of different signatures or to specialize a signature. A detailed discussion of SHEDEL and example signatures are contained in [18].

4.2 An algorithm for event-based misuse detection

Proceeding from signatures described in SHEDEL, an algorithm is needed, which matches the specified patterns against a stream of audit data. Because of the reasons discussed in Section 2.3 neither a simple code generation approach nor a (general-purpose) expert system shell is used for this purpose. Instead we develop a special-purpose algorithm, named StraFER (*Straight-Forward Event Recognition*), that is based on descriptions in SHEDEL. In this section we first present a straight-forward algorithm for matching event-based signatures. Second we discuss some ideas for optimizations of such an algorithm.

4.2.1 A STRAIGHT-FORWARD APPROACH FOR EVENT-BASED MISUSE DETECTION

A stream of audit records and a set of SHEDEL event descriptions form the input for the algorithm. Audit records are treated as observable input events. They are represented by SHEDEL basic events.

The analysis unit first reads all specified event descriptions and creates for each description a corresponding event object (an initial instance of the event description). During processing the straight-forward analysis algorithm does the following for each input event:

For each event object E

1. Check
   - if there are steps S of the type of the input event,
   - if all required predecessor steps P of these steps S have already occurred,
   - if all conditions, which refer to features of steps S respectively features of the input event and can already be evaluated, are fulfilled.
2. If these checks were successful
   - a copy of the event object E is created,
   - required information on the step S are saved in the copied event object,
   - the actions linked to step S are executed.
3. If step S is the last (and completing) step of event object E
   - the features of this event are assigned,
   - the occurrence of the event represented by event object E is signaled,
   - the event object is removed.

Occurred events are used as input event in the next cycle. After all occurred events are processed as input events the next audit record is used as input event.

A basic version of the StraFER algorithm is implanted. It detects all attack patterns specified as event descriptions; but it consumes a relatively high amount of memory and processing time.

4.2.2 IDEAS FOR OPTIMIZING EVENT-BASED ANALYSIS

The StraFER algorithm as discussed above checks all conditions of all event objects. Each of these checks consumes time. The runtime can be decreased

- by not checking each event object in every cycle,
- by not checking all conditions of each event object, and
- by optimizing the condition evaluation, for instance, by avoiding multiple calculations of the same function with the same parameters.

StraFER tries to realize the above points by taking advantage of knowledge about the structure and contents of its input data, especially the event based signature descriptions. Further some state information is maintained during the analysis to reduce the process runtime.

StraFER currently uses the following ideas:

1. Using the type of an input event the set of event objects, which have to be checked, is reduced to those event objects that contain a step of this type.
2. Steps of an event object usually are in temporal relation to each other. Thus the set of event objects that have to be checked can be reduced to those event objects with no or already occurred predecessor steps.
3. Events that represent only a partial signature (and thus have not linked an action to its final step) can be treated in a special way. Event `B` in Figure 5 shall be such an event. The occurrence of the final step `B3` of event `B` can imply the occurrence of event `B`. Event `B` is used as step in event `E` and has an event `A` as its predecessor step. As long as the event `A` does not have occurred the event `B3` does not have any effect with respect to the detection of event `E`. In this cases events like `B3` are called events with no effect. Such steps and the conditions associated with them do not have to be evaluated.

4. Conditions of an event object that describe relations between features of different steps have only to be evaluated if they
   - refer to features of steps with a type equal to the type of the current input event and
   - use to features of already occurred steps.

The ideas mentioned in point 1 and 2 reduce the number of event objects. Additionally the number of conditions is decreased using the concepts in point 3 and 4.

To implement the above ideas different calculations are required. A main part of them can be done at compile time (during the reading of the event descriptions) using information about the static structure of signatures. During runtime only a few modifications of control data are needed. Thus a noticeably runtime improvement can be expected. We are currently introduce these optimizations into the StraFER implementation.

5 Conclusion

In the paper we have presented the basics of the intrusion detection infrastructure HEIDI. The HEIDI approach aims at a module system to set up efficient and tailored intrusion detection systems for local area networks. The module system provides a set of specialized sensors for audit data capturing and flexible agents for data analysis. The analysis combines local and central signature matching processes. Furthermore, a HEIDI-based intrusion detection system is capable to react to overload situations by delegating analysis functionality among the communicating agents.

So far only a very few intrusion detection approaches or systems have the potential to overcome the efficiency problem of the host-oriented intrusion detection. It has shown that only decentralized analysis approaches like DIDS, AAFID, and EMERALD are capable to meet near real-time requirements. Unfortunately, none of these systems aim at an efficient intrusion detection solution, but from an architectural point of view some aspects are comparable with HEIDI. The systems CSM and EMERALD are characterized by the application of large and complex modules. This feature is similar to the HEIDI agents. In contrast to this AAFID uses a great number of small an specialized entities. In HEIDI the sensors play a similar role. Since HEIDI uses both complex and small modules it is also comparable to DIDS. Depending on the different development targets each of this systems has special module-intern structures. Regarding the module interconnection capabilities HEIDI seems to be as potentially as EMERALD and AAFID, while DIDS and CSM are functionally limited in this context.

The implementation of the HEIDI infrastructure modules is still in progress. Currently the implementation of various sensors, e.g. for capturing audit data under Sun Solaris and Microsoft Windows NT/2000, and for TCP/IP segments are available. After finishing the implementation of the HEIDI agent we plan to set up a first example intrusion detection system which corresponds our system AID [4]. By comparing the two AID variants we will evaluate the performance of the HEIDI concept. Thereafter the usability of the HEIDI concept will be investigated with different IDS architectures.

To support efficient data analysis with HEIDI agents the new algorithm StraFER that utilize knowledge about the static structure of signatures has been proposed. A basic version of StraFER is available. The next step will be to evaluate the impact of the discussed optimizations on the processing time. We plan to directly compare the StraFER implementation with the rule based analysis component of AID. Based on this further optimizations will be introduced to the algorithm.

References

[1] Clinton Administration (ed.): The Clinton Administration's Policy on Critical Infrastructure Protection : Presidental Decision Directive 63, 1998.

[2] Denning, Dorothy E.: Information Warfare and Security. Addison Wesley Longman, Inc., Reading, 1999.

[3] Meier, Michael; Holz, Thomas: Intrusion Detection Systems List and Bibliography. http://www-rnks.informatik.tu-cottbus.de/en/security/ids.html, 2001.

[4] Sobirey, Michael; Richter, Birk; Koenig, Hartmut: The Intrusion Detection System AID - Architecture, and experiences in automated audit analysis. In: Horster, Patrick (ed.): Proceedings of the IFIP TC6/TC11 International Conference on Communications and Multimedia Security at Essen, Germany, 23rd - 24th September 1996. Chapman & Hall, London, 1996, pp. 278-290.

[5] Axelsson, Stefan: Research in Intrusion Detection Systems: A Survey. Goeteborg, Chalmers University of Technology, Technical Report No. 98-17, 1998.

[6] Snapp, Steven R.; Smaha, Stephen E.; Teal, Daniel M.; Grance, Tim: The DIDS (distributed intrusion detection system) prototype. In: USENIX Association (ed.): Proceedings of the Summer 1992 USENIX Conference. USENIX Association, Berkeley, 1992, pp. 227-233.

[7] White, Gregory B.; Pooch, Udo W.: Cooperating security managers: Distributed intrusion detection systems. In: Computers & Security 15 (1996), No. 5, pp. 441-450.

[8] Spafford, Eugene H.; Zamboni, Diego: Intrusion detection using autonomous agents. In: Computer Networks 34 (2000), No. 4, pp. 547-570.

[9] Porras, Phillip A.; Neumann, Peter G.: EMERALD: Event Monitoring Enabling Responses to Anomalous Live Disturbances. In: NIST (ed.); NCSC of the NSA (ed..): Proceedings of the 20th NISSC, 1997. National Institute of Standards and Technology, Gaithersburg, 1997, pp. 353-365.

[10] Mounji, Abdelaziz: Languages and Tools for Rule-Based Distributed Intrusion Detection. University of Namur, Belgium, Computer Security Institute, PhD Thesis, 1997.

[11] Vigna, Giovanni; Eckmann, Steven T.; Kemmerer, Richard A.: The STAT Tool Suite. In: Proceedings of DISCEX, Hilton Head, South Carolina. IEEE Computer Society Press, Los Alamitos, California, 2000.

[12] Kumar, Sandeep: Classification and Detection of Computer Intrusions. Purdue University, PhD Thesis, 1995.

[13] Lindqvist, Ulf; Porras, Phillip A.: Detecting Computer and Network Misuse with the Production-Based Expert System Toolset (P-BEST). In: Proceedings of the IEEE Symposium on Security and Privacy. IEEE Computer Society Press, Los Alamitos, California, 1999, pp. 146-161.

[14] Forgy, Ch. L.: Rete: A Fast Algorithm for the Many Pattern / Many Object Pattern Match Problem. In: Artificial Intelligence, No. 10, 1982, pp. 17-37.

[15] Miranker, Daniel P.: TREAT: A better match algorithm for AI production systems. In: Proceedings of AAAI 87 Conference on Artificial Intelligence, 1987, pp. 42-47.

[16] Miranker, Daniel P.; Brant, D. A.: An Algorithmic Basis for Integrating Production Systems and Large Databases. In: Proceedings of the Sixth International Conference on Data Engineering, February 5-9, 1990, Los Angeles, California, USA. IEEE Computer Society, 1990, pp. 353-360.

[17] Wang, Y.; Hanson, E. N.: A Performance Comparison of the Rete and TREAT Algorithms for Testing Database Rule Conditions. In: Golshani, Forouzan (ed.): Proceedings of the Eighth International Conference on Data Engineering, 1992. IEEE Computer Society, 1992, pp. 88-97.

[18] Meier, Michael; Bischof, Niels; Holz, Thomas: SHEDEL - A Simple Hierarchical Event Description Language for Specifying Attack Signatures. To appear in: Proceedings of the IFIP International Conference on Information Security ,Cairo, Egypt, 2002.

Click here to view PowerPoint presentation; Press Esc to exit

Anomaly Detection for Multimedia Traffic

Ralf Wienzek, Mark Borning
Aachen University of Technology – Department of Computer Science
Informatik 4 (Communication Systems)
52056 Aachen, Germany

Phone: +49 (241) 80-21412, Fax: +49 (241) 80-22220
{wienzek,borning}@i4.informatik.rwth-aachen.de

Roland Büschkes
T-Mobile Deutschland GmbH
POB 300463, 53184 Bonn, Germany

Phone: +49 (228) 936-3485, Fax: +49 (228) 936-3309
Roland.Bueschkes@t-mobile.de

Abstract

In this paper we present attacks against multimedia streams and their underlying protocols and architectures. In order to describe these attacks the behavior of data sources and networks is modeled and an anomaly detection approach is chosen to detect them. The used methods are analyzed by their false alarm rate and the amount of time needed to detect an on-going attack.

1 Introduction

Today's Internet offers a *Best Effort Service*, i.e packets are transmitted without any service guarantees. For services like WWW, email, chat, etc. this behavior is sufficient. But during the last years more and more applications transmitting multimedia data (audio, video) have been developed and are in use today. They offer users high-quality services like Internet telephony (Voice over IP, VoIP) or video conferencing.

Generally, these applications make higher demands on the transmission quality than conventional services. Therefore, new protocols have been designed in order to guarantee the required qualities. Due to the related increase of complexity, new forms of attacks are possible and, hence, new techniques for their prevention or at least their detection are required.

The paper is divided into six parts. After this introduction, protocols and architectures designed to offer the required quality of service are described. In section 3 the techniques used by intrusion detection systems are resumed and possible attacks against multimedia applications and architectures are presented. Thereafter, we present our modeling of objects involved in communication and the corresponding detection mechanisms, which are evaluated in section 5. Section 6 draws some conclusions and closes the discussion with an outlook onto future work.

2 Protocols and Architectures

Multimedia applications have to process periodically sampled data like audio or video signals. The resulting data stream is encoded, compressed, divided into packets of proper size and transmitted to the receiving application, which tries to reconstruct the original signal. In order to ensure a good reconstruction quality, multimedia applications require more service guarantees (*Quality of Service, QoS*) than offered by Best Effort. The ITU standard X.902 defines QoS as a set of quality requirements on the collective behavior of one or more objects. In general, there are many parameters which can be used to describe QoS. For the purpose of this paper the following parameters are important:

1. **Throughput:** Measures the number of bytes of user data transferred per second. Depending on the codec used for data encoding the needed throughput varies from about 5.3 kbps for audio (G.723.1)

Paper presented at the RTO IST Symposium on "Real Time Intrusion Detection", held in Estoril, Portugal from 27-28 May 2002, and published in RTO-MP-101.

over a multiple of 64 kbps for video conferencing (H.261, H.263) up to several Mbps for MPEG transmissions.

2. **Transit delay:** Measures the time between a packet being sent by the peer entity on the source host and the receiving of the packet by the peer entity on the destination host. The required upper bound for this delay is subjective. For some users a delay of 200 ms is too much, whereas for others a delay of up to 500 ms is still acceptable [KBS98]. In addition to the mentioned delay, it is also important that all packets are delayed uniformly so that the playout buffer of the receiving application can be dimensioned accurately. Therefore, the variation of the packet's interarrival time, the so-called *jitter*, should be as small as possible.
3. **Packet loss rate:** Measures the number of lost or garbled packets as a fraction of the total sent. The effect of packet loss is application dependent. Some codecs use for example a forward error correction algorithm in order to compensate packet losses. Furthermore, the distribution of packet losses is important. Normally, the loss of many consecutive packets has a more significant effect than an uniformly distributed one, although in both cases the resulting loss rate may be equal.

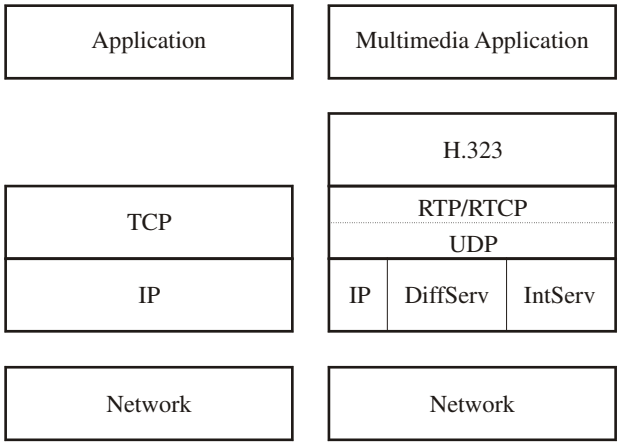


Figure 1: Protocol overview.

In order to provide the required quality of service guarantees the Internet's original TCP/IP stack has been extended. In Fig. 1 the protocol stack for conventional applications (left) is compared to the one used by a multimedia application (right). In the conventional case, TCP data is sent within IP packets, which can be transmitted over several intermediate networks. For these kind of connections it is important that data is transferred completely and correctly. The packet delay and the distribution of packet delays are of less importance.

For multimedia applications this is no longer the case. A low loss rate is usually acceptable and due to the human ability to compensate playback errors it is not essential to have an absolute error free connection. However, it is important that packets are received early enough to meet their playout time. Packets arriving too late have to be discarded because their data has become useless. Furthermore, since there are a variety of possible codecs used by multimedia applications it has to be determined which codec should be used and how it should be parameterized. Consequently, an additional signaling protocol is needed to handle this problem. On the right side of Fig. 1 a possible protocol stack is shown. Multimedia applications use H.323 to set up and parameterize a session. The generated data is sent using RTP which often uses UDP as the underlying transport protocol. Differentiated Services (DiffServ) and Integrated Services (IntServ) are two architectures developed to offer a certain kind of QoS to data streams.

In the following subsections, RTP, DiffServ and IntServ are described briefly.

2.1 RTP/RTCP

The *Real-time Transport Protocol (RTP)* [RFC1889] has been developed as a transport protocol for real-time data. It consists of two parts: 1) RTP to carry real-time data and 2) the *RTP Control Protocol (RTCP)* to monitor the QoS and to convey information about the participants of an on-going session. RTP has been designed to be used in multicast sessions, i.e. all participants subscribe to the same multicast address (determined for instance using H.323) and send and/or receive one or more data streams. In order

to be able to identify packets of a certain stream, a randomly chosen but session-wide unique identifier is attached to each stream (the so-called synchronization source ID, SSRC-ID). Furthermore, each RTP packet contains a sequence number, which is used to detect packet losses. Moreover, RTP offers a timestamp field describing the sampling time of the first data byte. The resolution and frequency of the used clock is application dependent. A further feature is a marker bit used to signal important events.

The associated control protocol RTCP enables applications to compile statistics on the transmission quality. For this purpose, each participant regularly sends so-called *receiver reports* to all session members containing information about all data streams it is receiving. Among other things, these packets contain the loss rate, the highest sequence number received so far, and the measured interarrival jitter for each stream. In addition to the receiver reports, senders transmit so-called *sender reports* containing a NTP timestamp and the corresponding RTP timestamp of their generation, and the number of packets sent so far. With this information, a participant is able to evaluate the compliance of specified QoS parameters. Furthermore, RTCP is responsible for announcing the SSRC-ID of a specific data source.

2.2 Differentiated Services

DiffServ [RFC2475] is designed to be an architecture providing scalable service differentiation in the Internet. A service level agreement (SLA) between a customer and a service provider is concluded specifying the forwarding service the customer should receive. Within a DiffServ network, two kinds of nodes exist: 1) DS boundary nodes connecting one DS domain[1] to a node either in another DS domain or in a domain that is not DS-capable and 2) DS interior nodes which are DS nodes but no boundary nodes. Each incoming packet is processed by a boundary node depending on the SLA and the so far measured sending characteristic of the stream it belongs to. In Fig. 2 a logical view of a boundary node is shown.

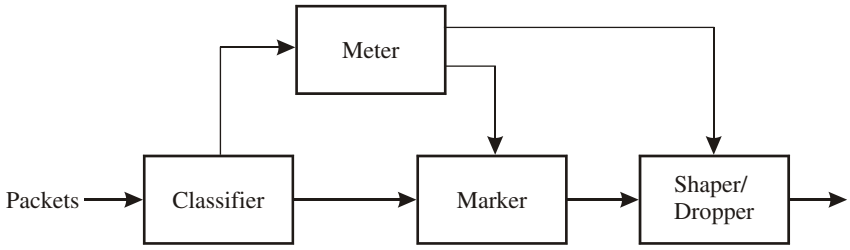


Figure 2: Logical view of a packet classifier and traffic conditioner.

Classifiers select packets in a traffic stream based on the content of some portion of the packet header. They are used to direct packets matching some specified rule to an element of a traffic conditioner for further processing. The traffic conditioner may contain meters, markers, shapers, and droppers. The meters are used to measure the traffic stream against a traffic profile. The state of the meter with respect to a particular packet (e.g., whether it is in- or out-of-profile) may be used to affect a marking, dropping, or shaping action. For each packet, markers set the DS codepoint used by the interior nodes to forward the packet according to its SLA. Shapers delay and droppers discard some or all packets of a data stream in order to make the stream compatible with its traffic profile.

In order to provide a specific service for a customer, a so-called *per hop behavior* is implemented in each interior node. All interior nodes of a DS domain agree to treat a packet with a specific DS codepoint in the same way. If, for instance, the DS codepoints represent levels of drop precedence [RFC2597], it is possible to guarantee a deliverance of all packets meeting their SLA, whereas for packets, which are out of the limits given by the SLA, it is more likely to get lost if the network is congested.

[1] A DS domain is a contiguous set of network nodes which operate with a common set of service provisioning policies.

2.3 Integrated Services and RSVP

The IntServ architecture [RFC1633] is designed to extend the Internet to provide applications their needed QoS. IntServ requires the explicit management of all resources, i.e. resources can be reserved for a data stream and used exclusively. In addition to the Best Effort Service, IntServ provides among others two services:

- *Guaranteed Service*: A perfectly reliable upper bound on delay is given and, furthermore, the guarantee of no packet losses. Perfectly reliable means that no single packet is delayed longer than the specified upper bound.
- *Predictive Service*: Predictive service supplies a fairly reliable, but not perfectly reliable, delay bound. There may be a small fraction of packets with a higher delay than the given bound, but the great majority of all packet delays are within this limit.

In order to be able to reserve the required resources, the *Resource Reservation Protocol* (*RSVP*) [RFC2210] has been defined. It is used by possible receivers to signal the desired quality and by IntServ switches to promote the request to the data source. Therefore, all session members subscribe to the same multicast group and hosts with data sources regularly send so-called *path messages* providing information about the streams they are sending or are able to send. When a receiver receives a path message for a data source it is interested in, it sends a *reservation message* back to the source. This message contains a so-called *flowspec* with information about the desired quality. This message is guided along the reverse route of the corresponding path message and every involved node checks whether it has enough free resources for this reservation. If any node rejects the reservation, the receiver is informed and the reservation message is discarded. In [RFC1363] an example of a flowspec is given which uses a token bucket for the description of a data source. Furthermore, an upper bound for the variation in end-to-end transmission delays and the sensitivity to data loss are specified. Having received a reservation message, the sender starts sending data packets with the requested quality. Data packets take the reverse path of the reservation message and use the reserved resources along this path. The reservations in the nodes are valid only for a small period (*soft states*), so that receivers have to periodically send reservation refresh messages, which are identical in format to the original request.

3 Intrusion Detection Systems

Different kinds of attacks against communication networks are known. However, they can be subdivided into three categories:

1. Attacks against confidentiality: The attacker tries to gain information he is not allowed to see. This refers to both, contents of packets and the existence of a communication relationship.
2. Attacks against integrity: The attacker tries to modify the content of one or more packets.
3. Attacks against availability: The attacker tries to prevent authorized users to access resources or tries to delay time critical operations [Gol99]. These attacks are called *Denial-of-Service* (DoS) attacks.

Attacks of the first two categories can be prevented by using conventional methods like encryption to ensure confidentiality (e.g. IPsec [RFC2401]) and message digests to ensure integrity (e.g. MD5). However, a prevention of DoS attacks is difficult to realize in today's networks. To be at least able to detect DoS attacks, intrusion detection systems (IDS) are used.

The techniques used by intrusion detection systems can be subdivided into two categories: misuse detection and anomaly detection. Misuse detection tries to detect patterns of known attacks within the audit stream of a system, i.e. it identifies attacks directly. In order to do so it explicitly specifies attack patterns and monitors the audit stream for any occurrences of these patterns. The dual approach is the specification of the desired behavior of the monitored objects. Attacks are identified by observing deviations from this norm. Therefore, this technique is called anomaly detection.

Two general approaches exist to specify the normal behavior: 1) learning and 2) specification of behavior. The first approach is often based on statistical methods like the calculation of means, variations and multivariate statistics. Other methods use learning algorithms like e.g. neural networks. The second approach, specification-based anomaly detection, is based on the formal description of the desired

behavior. It was first proposed by Calvin Ko [Ko96] and has recently been followed by other works [SBS99,BNB01].

By using for instance Transaction-Based Anomaly Detection [BNB01,Büs01] the object's normal behavior is specified by deterministic finite state machines. Typical DoS attacks like SYN flood, i.e. the third packet of the TCP three way handshake is not sent for several connections, and the ping of death, i.e. sending a fragmented ICMP packet of an improper size, can be detected; the first one by a violation of atomicity (the final state of the state machine is not reached) and the second by a violation of consistency (the transition function for this input is not specified).

As mentioned in the previous section, multimedia traffic needs QoS, i.e. not only the arrival of packets is important but also their correct timing. Consequentially, new DoS attacks are possible, which are not detected explicitly by a conventional IDS.

3.1 Attacks against multimedia streams

In the following, possible attacks against multimedia streams and protocols are presented. They can be originated by either the communication partner or by a third party controlling one or more network elements.

- The sending rate of RTP data packets is too high. The attacker tries to overload the receiver or the network. This attack has to be detected by a statistical analysis of the sending characteristic.
- No data or data of lower quality as desired by the receiver is sent. The attacker tries to prevent the receiver from the quality of service it is allowed to get. This attack can be detected by a statistical analysis, too, or, in case of no data arriving at all, by a violation of atomicity [Büs01].
- RTP packets are delayed by an attacker. The effect of this attack is application dependent. A uniform delay of all packets does not significantly affect an application without interactivity (e.g. video on demand). Irregular delays lead to a high interarrival jitter, so that some packets miss their playout time or they arrive too early to be buffered by the receiver. In the worst case, the replay quality is unacceptable (DoS). Interactive applications (e.g. telephony) are definitely affected by delaying data packets.
- RTP packets are discarded by network nodes. The effect of this attack depends on the amount and distribution of data losses and the use of redundant information in order to reconstruct missing data. If all data packets are discarded, a violation of atomicity can be detected. Otherwise, packet loss behavior can be determined by interpreting the sequence number contained in RTP packets.
- Not all or no RTCP sender or receiver reports are sent or they are discarded by network nodes. The attacker tries to prevent the session's participants from determining the current transmission quality. A too positive estimation of the current quality can give an adaptive sender occasion to increase its transmission quality and lead to a network congestion. On the other hand, a too negative estimation can give occasion to unnecessarily decrease the transmission quality. Furthermore, by means of RTCP packets, the number of participants is determined on which the sending rate of RTCP packets depends. The more participants a session has the lower the sending rate has to be. RTCP packets don't contain sequence numbers, i.e. this attack can only be detected by statistically analyzing the RTCP sending behavior or, if no packet arrives, by a violation of atomicity.
- The RTCP packet sending rate is too high or too low. As mentioned above, a high sending rate can cause a network congestion whereas a low sending rate can result in inexact estimations of the current network situation.
- The sending rate of RSVP path messages or reservation messages is manipulated. The increase of a sending rate can cause a network congestion. If the path message's sending rate is decreased, potential receivers have to wait unnecessarily long for information about offered data streams. The decrease of reservation messages can result in a reset of RSVP soft states so that resource reservations for data packets do not exist any longer.
- All path messages are discarded or no path messages are sent. This means a denial of service, since, due to the lack of information about available data streams, receivers are not able to make reservations. If all reservation messages are blocked by an attacker, the sender is not informed about

the receiver's wish to receive its data and therefore does not start to send any. If the sender knows by another source that a specific receiver is interested in its data, this attack can be detected by a violation of atomicity; otherwise it is only detectable by a distributed anomaly detection.

- Within a DiffServ network, the work of shapers and droppers is manipulated or the DS codepoint of one or all packets is changed. These attacks result in modified characteristics of the involved data streams and can be detected by a statistical analysis of incoming data.

Based on the given examples two categories of attacks can be identified:

1. Attacks detectable by a violation of atomicity in the model of transaction based anomaly detection.
2. Attacks detectable by a statistical analysis of the involved data streams.

The attacks of the second category can be abstracted to one of the following three abstract attacks:

1. Attacks manipulating the sending characteristic. This can mean both an increase and a decrease. This abstract attack represents for instance the sending of RTP packets with an inaccurate rate, the manipulation of the RTCP sending rate or the sending of reservation messages with a too low rate.
2. Attacks manipulating the network's loss behavior. This abstract attack represents all attacks trying to discard packets (deterministically or randomly) in order to decrease the transmission quality.
3. Attacks manipulating the transmission delay. On the one hand this abstract attack encloses the increase of the average delay and on the other hand the increase of the interarrival jitter.

In the following section, models describing these attacks are introduced and subsequently methods for their detection are presented.

4 Model and Detection

In this section, formal models for the description of data sources and the underlying network are given, followed by an introduction of methods used to detect attacks against them. Therefore, two forms of data sources can be distinguished: constant bit rate (CBR) and variable bit rate (VBR). Their data characteristics differ significantly, so that they have to be considered separately.

4.1 CBR source

Codec	Rate [kbps]	Frame duration [ms]	Frame size [byte]
G.723.1	5.3/6.4	30	20/24
G.729	8	10	10

Table 1: Characteristics of two audio codecs.

If a source regularly sends equally sized packets, it is called a CBR source. In Tab. 1 the characteristics of two audio codecs are given. It can be seen, that, for instance in the case of G.729, a packet of 10 bytes is generated every 10 ms. Therefore, two parameters are needed to describe a CBR source: the time between two packets δ and the packet size s.

In order to detect anomalies of a CBR source's behavior, an IDS has to compare the measured packet size $\hat{s}$ of incoming packets to the specified packet size s. Additionally, the measured time $\hat{\delta}$ between two packets has to satisfy $\delta \approx_\varepsilon \hat{\delta}$, which is an abbreviation of $\delta - \varepsilon \le \hat{\delta} \le \delta + \varepsilon$, with $\varepsilon \ge 0$ specified by the IDS. If one of these conditions is violated, an anomaly is detected.

In addition to sources permanently sending with a CBR characteristic, there are also sources using silence suppression algorithms in order to decrease the amount of transmitted data. These algorithms don't generate any data, while the communicating persons don't speak. So, a source with silence suppression is marked by alternating talk spurts and silence gaps. During talk spurts it sends with CBR characteristic and during silence gaps no data is transmitted. In former silence suppression models the length of both talk spurts and silence gaps is assumed to be exponentially distributed with a mean somewhere in the order of 1 second [Bra69]. Although it is in question, whether this assumption still holds for today's

algorithms [JS00a], it should be an accurate approximation and is therefore used in the following. So, in order to describe a CBR source with silence suppression two more parameters are needed: μ_S and μ_G specifying the average length of a talk spurt and a silence gap resp.

Anomaly detection becomes a bit more complicated, too. Additionally to the above mentioned checks for $\hat{s}$ and $\hat{\delta}$, the average lengths of the measured talk spurts $\hat{\mu}_S$ and silence gaps $\hat{\mu}_G$ have to be determined and compared to the specified ones, i.e. the conditions $\mu_S \approx_{\varepsilon_1} \hat{\mu}_S$ and $\mu_G \approx_{\varepsilon_2} \hat{\mu}_G$ have to hold (with $\varepsilon_1, \varepsilon_2 \geq 0$ specified by the IDS).

Two methods for computing the average $\hat{\mu}_t$ after the evaluation of the *t*-th value of an arbitrary long sequence of values $x_1, x_2, ...$ shall be considered in this paper. These are

1. maintaining a ring buffer of size *N*, containing the last *N* values, and computing $\hat{\mu}_t$ as
   $\hat{\mu}_t = \frac{1}{N}\sum_{i=0}^{N-1} x_{t-i}$, and (1)
2. computing $\hat{\mu}_t$ as an exponential weighted moving average (EWMA), i.e. as a weighted sum of the last measured value x_t and the last computed $\hat{\mu}_{t-1}$:
   $\hat{\mu}_t = \alpha x_t + (1-\alpha)\hat{\mu}_{t-1}$, (2)
   with $\alpha \in [0,1]$ indicating the weight of the last measured value.

4.2 VBR source

Even if the specification of CBR sources is simple, many sources do not have this behavior, since they irregularly send packets or their packets are not equally sized. These sorts of sources are called VBR sources. Video conferencing applications for example normally are VBR sources. Often, the generated stream is shaped by a *leaky bucket* or a *token bucket* [Tan96] in order to produce a more predictable data stream. The flowspec for RSVP described in [RFC1363] for example uses a token bucket to specify the behavior of a data stream. In order to be able to detect anomalies, it is not sufficient to just verify that a stream is not exceeding a certain rate. Since a possible attack against a multimedia application is the reduction of the data rate, it has also to be verified that a minimal amount of data arrives. Therefore, our VBR model uses both: a token bucket to define an upper bound for the amount of data and a leaky bucket to define a lower bound.

Provided that the data stream is shaped by a token bucket of size *B* and rate *r*, the IDS checks whether the monitored data stream behaves accordingly. For this, a token bucket of size *B* is used, which is initially filled. Let a_t denote the sending time and s_t the size of the *t*-th packet. From the last packet up to this one $(a_t - a_{t-1})r$ tokens are filled into the bucket so that its fill level B_t at the moment the packet is sent is given by $B_t = \min\{B_{t-1} + (a_t - a_{t-1})r, B\}$. To be conforming to the specification, enough tokens have to be in the bucket, i.e. $s_t \leq B_t$ must hold. If this is not case, an anomaly is detected. Otherwise, the fill level has to be updated by subtracting s_t from B_t.

Similarly, the conformance with a minimal data rate is checked. For a short period of time, the sending rate should be allowed to fall below the minimal rate of *r*. Therefore, a leaky bucket of size *B* and rate *r* is used, which is initially filled with tokens. At time a_t, $(a_t - a_{t-1})r$ tokens have left the bucket since the last update, so that the fill level B_t is given by $B_t = B_{t-1} - (a_t - a_{t-1})r$. The smaller the value of B_t is the lower the sending rate has been in the near past. An anomaly is detected if $B_t < 0$. Otherwise, B_t is updated by adding s_t.

4.3 Packet loss

In order to evaluate the effect of packet losses, it is not sufficient to just measure the average loss rate, since several significant different situations can result in the same average loss rate. Therefore, packet loss is modeled by an extended Gilbert model [JS00b], which is a special case of a n-th order Markov chain. As depicted in Fig. 3, not all state transitions are admitted. The state q_i (*i=0,...,n-1*) represents the

loss of *i* consecutive packets. With probability p_i the next packet is also lost and with probability *1-*p_i it makes its way through the network. In the first case state q_{i+1} is entered, otherwise q_0, which is also the initial state. So, p_i describes the probability that a packet gets lost, if exactly i consecutive packets are lost.

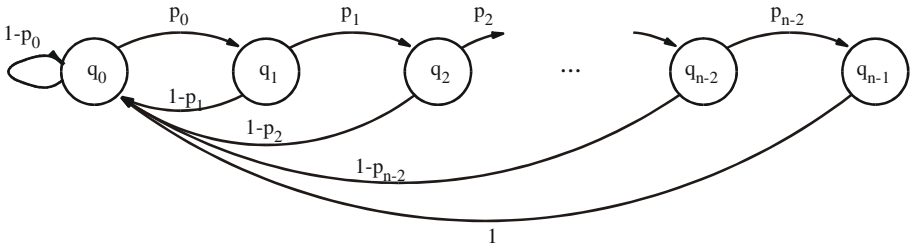


Figure 3: Extended Gilbert model.

In order to detect anomalies of the loss behavior, the frequency distribution $\hat{\pi}_i$ of the q_i's has to be measured from the data stream by

$$\hat{\pi}_i = \frac{\text{number of loss runs of length} \geq \text{i}}{\text{number of expected packets}}.$$

The number of expected packets can be determined by the greatest RTP sequence number received so far. The measured $\hat{\pi}_i$ may not significantly exceed the steady state probabilities π_i given by the model as

$$\pi_i = \pi_0 \prod_{k=0}^{i-1} p_k \text{ for } i=1,...,n-1 \text{ and } \pi_0 = \left(1 + \sum_{i=1}^{n-1} \prod_{k=0}^{i-1} p_k\right)^{-1}.$$

An anomaly is detected if an $i \in \{1,...,n-1\}$ exists so that $\hat{\pi}_i \leq \pi_i + \varepsilon_i$ with $\varepsilon_i \geq 0$ given by the IDS.

4.4 Packet delay

The simplest model of packet delay is an upper bound $d_{\max}$ indicating the maximal amount of time a packet may need to travel through the network (cp. IntServ guaranteed service). Anomaly detection is very simple, too. Given the delay $\hat{d}$ of the current packet, an anomaly occurs if $\hat{d} > d_{\max}$ holds.

A less restrictive delay model is given by a Gaussian distribution with parameters (μ,σ). Thereby, μ describes the average packet delay and σ the standard deviation corresponding to the interarrival jitter. In order to detect anomalies these two parameters have to be measured and compared to the specified ones.

The computation of the average values can be done by using either formula (1) or (2). An estimation of the variance after *t* packets can be computed from the last *N* packets by

$$\hat{\sigma}_t^{\,2} = \frac{1}{N-1} \sum_{i=0}^{N-1} (d_{t-i} - \hat{\mu}_t)^2 . \qquad (3)$$

The delay behaves normal if $0 \leq \hat{\mu}_t \leq \mu + \varepsilon_1$ and $0 \leq \hat{\sigma}_t^{\,2} \leq \sigma^2 + \varepsilon_2$ hold.

5 Performance Evaluation

In this section, the proposed methods for anomaly detection are analyzed. Some of those methods compare statistical parameters like mean and variance. Two methods for the average value computation of a sequence of numbers were given: using a ring buffer of size *N* (formula (1)) and EWMA with parameter α (formula (2)). For both methods, a possible parameterization is evaluated by the following two criterions:

1. The rate of false alarms. This value indicates the probability that an anomaly is reported, although the monitored objects behave according to their defined norms.

2. The amount of time elapsing until an on-going attack is detected, i.e. how many measures are necessary until a change of the characteristic is detected.

In the following subsection, the average spurt and gap length computation of a CBR source with silence suppression is analyzed followed by an evaluation of a Gaussian distributed packet delay. Finally, the effect of packet loss on measured data source parameters is considered.

5.1 Length of spurts and gaps of a CBR source with silence suppression

In order to monitor the average length of talk spurts and silence gaps, it is assumed, that those lengths are exponentially distributed with mean μ_S and μ_G resp. Hence, a sequence of independent and identical distributed (iid) exponential random variables X_i with expected value $E[X_i]=\mu$ ($i=1,2,\ldots$) has to be analyzed. The IDS checks, whether the measured mean $\hat{\mu}$ of the values x_i of those random variables X_i corresponds with the given value μ. For this, at each occurrence of an event x_t, the measured mean $\hat{\mu}_t$ is updated and the deviation from the norm μ is determined. If this deviation is too high, i.e. $|\hat{\mu}_t-\mu|>\varepsilon$, an anomaly is detected. The following analysis evaluates a given parameterization (N,ε) of the ring method and (α,ε) of the EWMA method according to the above-mentioned criterions.

5.1.1 RING METHOD

By using the ring method, the current mean $\hat{\mu}_t$ is computed as the arithmetic mean of the last N measurements, i.e.

$$\hat{\mu}_t=\frac{1}{N}\sum_{i=t-N+1}^{t}x_i \tag{4}$$

(the special case $t<N$ shall be neglected). Since all x_i are values of random variables X_i, $\hat{\mu}_t$ is a random variable, too, with expected value $E[\hat{\mu}_t]=\mu$ and variance

$$Var(\hat{\mu}_t)=Var\left(\frac{1}{N}\sum_{i=t-N+1}^{t}X_i\right)=\frac{1}{N^2}Var\left(\sum_{i=t-N+1}^{t}X_i\right)=\frac{1}{N^2}\sum_{i=t-N+1}^{t}Var(X_i)=\frac{1}{N^2}\sum_{i=t-N+1}^{t}\mu^2=\frac{\mu^2}{N}. \tag{5}$$

Basically, $\hat{\mu}_t$ is the sum of N iid exponential random variables with parameter $1/\mu$. This sum is known to be Erlang distributed with parameter $(N,1/\mu)$.

The error rate Φ_R is given by the probability that for the measured $\hat{\mu}_t$ the condition $|\hat{\mu}_t-\mu|>\varepsilon$ holds, so

$$\begin{aligned}\Phi_R(N,\varepsilon,\mu) &= \Pr[|\hat{\mu}_t-\mu|>\varepsilon \mid E[X_i]=\mu]=1-\Pr[\mu-\varepsilon\le\hat{\mu}_t\le\mu+\varepsilon \mid E[X_i]=\mu]\\ &=1-\Pr[\mu-\varepsilon\le\frac{1}{N}\sum_{i=t-N+1}^{t}X_i\le\mu+\varepsilon \mid E[X_i]=\mu]=1-\Pr[(\mu-\varepsilon)N\le\sum_{i=t-N+1}^{t}X_i\le(\mu+\varepsilon)N \mid E[X_i]=\mu]\\ &=1-[F_{Er(N,1/\mu)}((\mu+\varepsilon)N)-F_{Er(N,1/\mu)}((\mu-\varepsilon)N)],\end{aligned}$$

whereby $F_{Er(N,1/\mu)}$ denotes the distribution function of an Erlang distribution with parameter $(N,1/\mu)$. Fig. 4 shows $\Phi_R(N,\varepsilon,\mu)$ for $\mu=1000$. It can be seen, that for values of ε somewhere in the order of *150* an acceptable false alarm rate is achieved for nearly all possible ring parameterizations N.

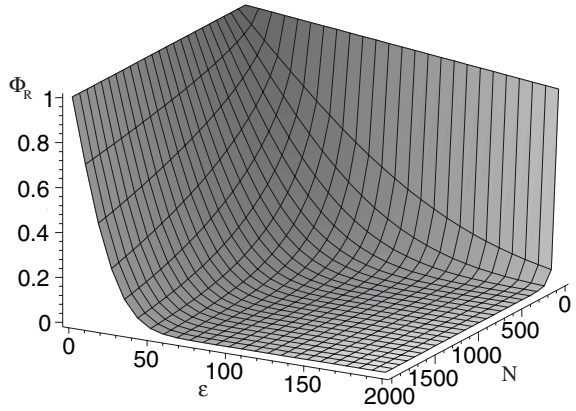


Figure 4: $\Phi_R(N,\varepsilon,\mu)$ **for** $\mu = 1000$**.**

In order to be able to evaluate the average number of measurements needed to detect a change of the sending behavior, the following setting is examined: After *t*>*N* exponentially distributed events $(X_1,...,X_t)$ with $E[X_i] = \mu$, $(i = 1,2,...,t)$, all following events $(X_{t+1}, X_{t+2},...)$ are exponentially distributed with $E[X_{t+j}] = \mu'$, $(j = 1,2,...)$ and $|\mu - \mu'| > \varepsilon$. The requested number of measurements $c_R(N,\varepsilon,\mu,\mu')$ is set to

$$c_R(N,\varepsilon,\mu,\mu') := n \text{ with } \begin{cases} E[\hat{\mu}_{t+n}] = \mu + \varepsilon, & \text{if } \mu' > \mu + \varepsilon \\ E[\hat{\mu}_{t+n}] = \mu - \varepsilon, & \text{if } \mu' < \mu - \varepsilon \end{cases}, \tag{6}$$

whereby $\hat{\mu}_i$ is the measured average after *i* events. The expected value $E[\hat{\mu}_{t+n}]$, $0 \le n \le N$, is given by

$$E[\hat{\mu}_{t+n}] = E\left[\frac{1}{N}\left(\sum_{i=-N+n+1}^{0} X_{t+i} + \sum_{i=1}^{n} X_{t+i}\right)\right] = \frac{1}{N}\left(\sum_{i=-N+n+1}^{0} E[X_{t+i}] + \sum_{i=1}^{n} E[X_{t+i}]\right) = \left(1 - \frac{n}{N}\right)\mu + \frac{n}{N}\mu' \tag{7}$$

which leads to $n = \dfrac{(E[\hat{\mu}_{t+n}] - \mu)N}{\mu' - \mu}$ and with the replacement of $E[\hat{\mu}_{t+n}]$ by $\mu + \varepsilon$ or $\mu - \varepsilon$ according to formula (6) to

$$c_R(N,\varepsilon,\mu,\mu') := \left|\frac{\varepsilon N}{\mu' - \mu}\right|.$$

The graphs of c_R for $\mu = 1000$ and $\mu' \in \{500, 2000\}$ are shown in Fig. 5.

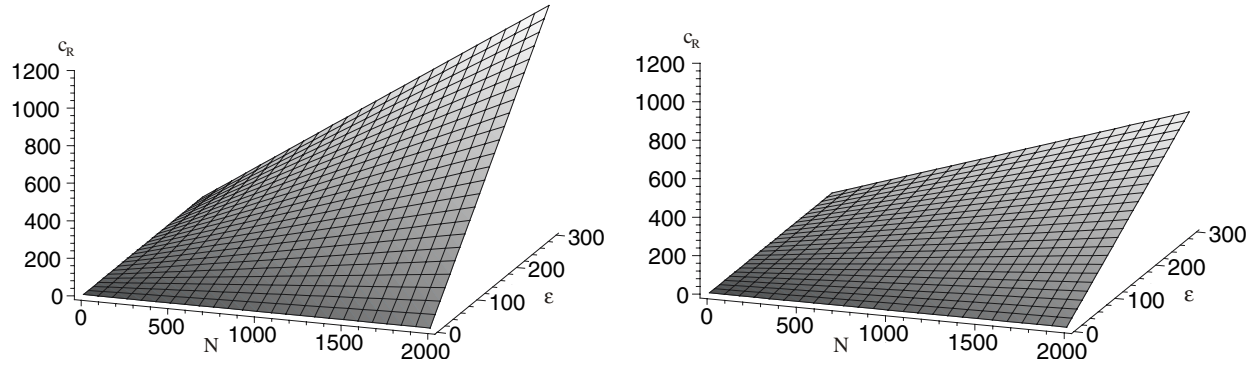


Figure 5: $c_R(N,\varepsilon,\mu,\mu')$ **for** $\mu = 1000$**,** $\mu' = 500$ **(left) and** $\mu' = 2000$ **(right).**

5.1.2 EWMA

The current average value measured by the IDS when using the EWMA method is given by

$$\hat{\mu}_t = \alpha X_t + (1-\alpha)\hat{\mu}_{t-1} = \begin{cases} \hat{\mu}_0 & \text{if } t = 0 \\ \alpha \sum_{i=0}^{t-1} (1-\alpha)^i X_{t-i} & \text{if } t > 0 \end{cases}.$$

By setting $\hat{\mu}_0 = \mu$ the expected value $E[\hat{\mu}_t]$ computes to μ. The variance of $\hat{\mu}_t$ is given by

$$Var(\hat{\mu}_t) = \mu^2 \frac{\alpha}{2-\alpha}(1-(1-\alpha)^{2t}) \xrightarrow[t\to\infty]{} \mu^2 \frac{\alpha}{(2-\alpha)}. \tag{8}$$

In order to be able to analytically describe the false alarm rate Φ_E of an EWMA parameterization, the following simplifying consideration is made. It is assumed that, if two random variables have both the same mean and variance, their false alarm rate is equal, too. Hence, for an EWMA parameterization (α, ε) a ring parameterization (N_α, ε) is considered, so that in both cases the random variable $\hat{\mu}_t$ has the same expected value and variance. Since $E[\hat{\mu}_t]$ is equal for both methods, N_α only depends on the variance. According to formulas (5) and (8)

$$\frac{\mu^2}{N_\alpha} = \mu^2 \frac{\alpha}{2-\alpha} \Leftrightarrow N_\alpha = \frac{2-\alpha}{\alpha}$$

has to hold and therefore

$$\Phi_E(\alpha, \varepsilon, \mu) := \Phi_R\left(\frac{2-\alpha}{\alpha}, \varepsilon, \mu\right).$$

The number of measurements needed to detect a change of μ to μ' ($|\mu - \mu'| > \varepsilon$) is computed in the same way as for the ring buffer method, namely

$$c_E(\alpha, \varepsilon, \mu, \mu') := n \text{ with } \begin{cases} E[\hat{\mu}_{t+n}] = \mu + \varepsilon, & \text{if } \mu' > \mu + \varepsilon \\ E[\hat{\mu}_{t+n}] = \mu - \varepsilon, & \text{if } \mu' < \mu - \varepsilon \end{cases}.$$

In this case the expected value $E[\hat{\mu}_{t+n}]$ is given by

$$\begin{aligned} E[\hat{\mu}_{t+n}] &= E\left[\alpha \sum_{i=0}^{t+n-1} (1-\alpha)^i X_{t+n-i} + (1-\alpha)^{t+n} \hat{\mu}_0\right] \\ &= \alpha\left(\sum_{i=0}^{n-1} (1-\alpha)^i E[X_{t+n-i}] + \sum_{i=n}^{t+n-1} (1-\alpha)^i E[X_{t+n-i}]\right) + (1-\alpha)^{t+n} E[\hat{\mu}_0] \\ &= \alpha \sum_{i=0}^{n-1} (1-\alpha)^i \mu' + \alpha \sum_{i=n}^{t+n-1} (1-\alpha)^i \mu + (1-\alpha)^{t+n} \mu \\ &= \mu'(1-(1-\alpha)^n) + \mu(1-\alpha)^n \end{aligned}$$

which is equivalent to $n = \frac{\ln\left(-\frac{E[\hat{\mu}_{t+n}]-\mu'}{\mu'-\mu}\right)}{\ln(1-\alpha)}$ and, with the replacement of $E[\hat{\mu}_{t+n}]$ by $\mu \pm \varepsilon$, leads to

$$c_E(\alpha, \varepsilon, \mu, \mu') := \frac{\ln\left(1 - \left|\frac{\varepsilon}{\mu'-\mu}\right|\right)}{\ln(1-\alpha)}.$$

5.2 Gaussian distributed packet delay

In order to detect anomalies of the packet delay, both the average and the variance have to be measured and compared to the parameters describing the desired behavior. In the following, for both parameters the false alarm rate and the number of measurements needed to detect a change are evaluated.

5.2.1 AVERAGE PACKET DELAY

In this subsection, the average packet delay is evaluated using the ring method. Therefore, a sequence $(X_1, X_2, ...)$ of iid normal random variables with mean μ and variance σ^2 (abbr. $X_t \sim \aleph(\mu, \sigma^2)$, $t = 1,2,...$) is considered. The IDS checks whether the measured mean $\hat{\mu}$ is significantly higher than the desired one, i.e. whether

$$\hat{\mu} > \mu + \varepsilon \,. \tag{9}$$

For this, the measured average $\hat{\mu}_t$ is updated according to formula (4) every time a packet is received. Since the X_t are $\aleph(\mu, \sigma^2)$ distributed, $\hat{\mu}_t$ is known to be $\aleph(\mu, \sigma^2 / N)$ distributed. The false alarm rate is given by the probability, that for the measured $\hat{\mu}_t$ formula (9) holds, i.e.

$$\Phi_R^{\mu}(N, \varepsilon, \mu, \sigma^2) = \Pr[\hat{\mu}_t > \mu + \varepsilon \mid X_i \sim \aleph(\mu, \sigma^2)] = 1 - \Pr[\hat{\mu}_t \le \mu + \varepsilon \mid X_i \sim \aleph(\mu, \sigma^2)]$$

$$= 1 - \Pr[\hat{\mu}_t \le \mu + \varepsilon \mid \hat{\mu}_t \sim \aleph(\mu, \sigma^2 / N)] = 1 - F_{\aleph(\mu, \sigma^2/N)}(\mu + \varepsilon),$$

whereby $F_{\aleph(\mu,\sigma^2)}$ denotes the distribution function of a $\aleph(\mu, \sigma^2)$ distributed random variable.

The number of measurements c_R^{μ} needed to detect a change of the average μ to μ' ($\mu' > \mu + \varepsilon$) is determined in the same way as in the previous section. The following scenario is considered: After $t > N$ events $X_i \sim \aleph(\mu, \sigma^2)$ ($i = 1,2,...,t$) all following events X_{t+j} are $\aleph(\mu', \sigma^2)$ distributed ($j = 1,2,...$). So, c_R^{μ} is defined by

$$c_R^{\mu} := n \text{ with } E[\hat{\mu}_{t+n}] = \mu + \varepsilon$$

which by replacing $E[\hat{\mu}_{t+n}] = \left(1 - \frac{n}{N}\right)\mu + \frac{n}{N}\mu'$ (see formula (7)), leads to

$$c_R^{\mu}(N, \varepsilon, \mu, \mu') = \frac{\varepsilon N}{\mu' - \mu}.$$

The graphs of Φ_R^{μ} and c_R^{μ} are depicted in Fig. 6 for $\mu = 350$ and $\sigma = 100$. It is noticeable that the false alarm rate becomes already small for small values of N and ε. This is reasoned by the standard deviation $\sigma / \sqrt{N}$ of $\hat{\mu}_t$ which is already relatively small for small values of N so that the measured average nearly equals its real value.

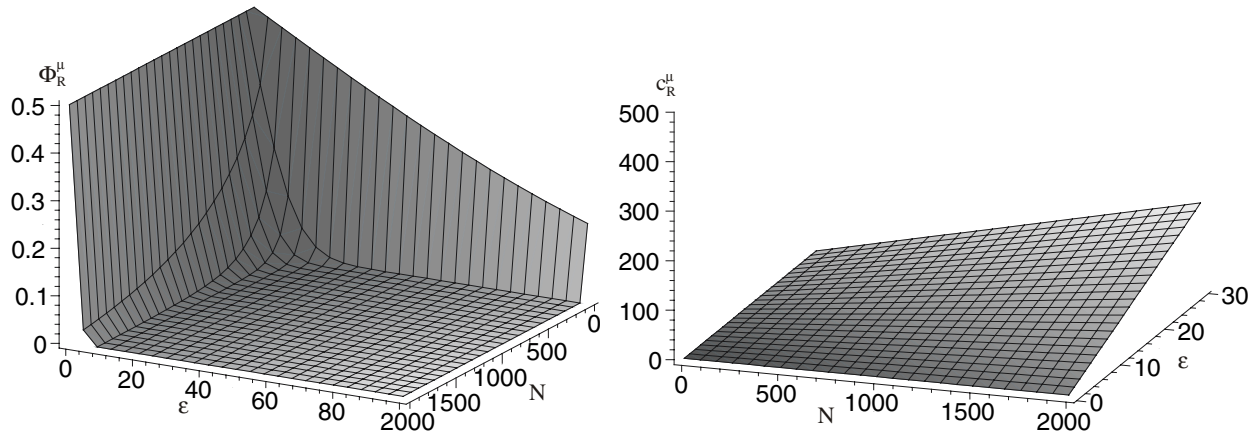


Figure 6: $\Phi_R^{\mu}(N, \varepsilon, \mu, \sigma^2)$ for $\mu = 350, \sigma = 100$ (left) and $c_R^{\mu}(N, \varepsilon, \mu, \mu')$ for $\mu = 350, \mu' = 700$ (right).

5.2.2 VARIANCE OF PACKET DELAY

In this subsection, the anomaly detection of the variance is evaluated. In this case the IDS checks whether for the estimated variance $\hat{\sigma}^2$ (see formula (3)) the condition

$$\hat{\sigma}^2 > \sigma^2 + \varepsilon$$

holds. Since $\hat{\sigma}^2$ is a composition of random variables, it itself is also a random variable known to be asymptotically $\aleph(\sigma^2, \frac{2}{N}\sigma^4)$ distributed. Therefore, the false alarm rate $\Phi_R^{\sigma^2}$ is determined by

$$\Phi_R^{\sigma^2}(N, \varepsilon, \sigma^2) = 1 - F_{\aleph(\sigma^2, \frac{2}{N}\sigma^4)}(\sigma^2 + \varepsilon).$$

The number of measurements $c_R^{\sigma^2}$ needed to detect a change from σ^2 to σ'^2 ($\sigma'^2 > \sigma^2 + \varepsilon$) is determined analogously to c_R^{μ} and is given by

$$c_R^{\sigma^2}(N, \varepsilon, \sigma^2, \sigma'^2) = \frac{\varepsilon N}{\sigma'^2 - \sigma^2}.$$

The graphs of $\Phi_R^{\sigma^2}$ and $c_R^{\sigma^2}$ are depicted in Fig. 7. It can be seen that a practical value for ε lies in the order of *1500*. Please notice, that the high values of ε are reasoned by the fact, that the variances are compared, and not the standard deviations.

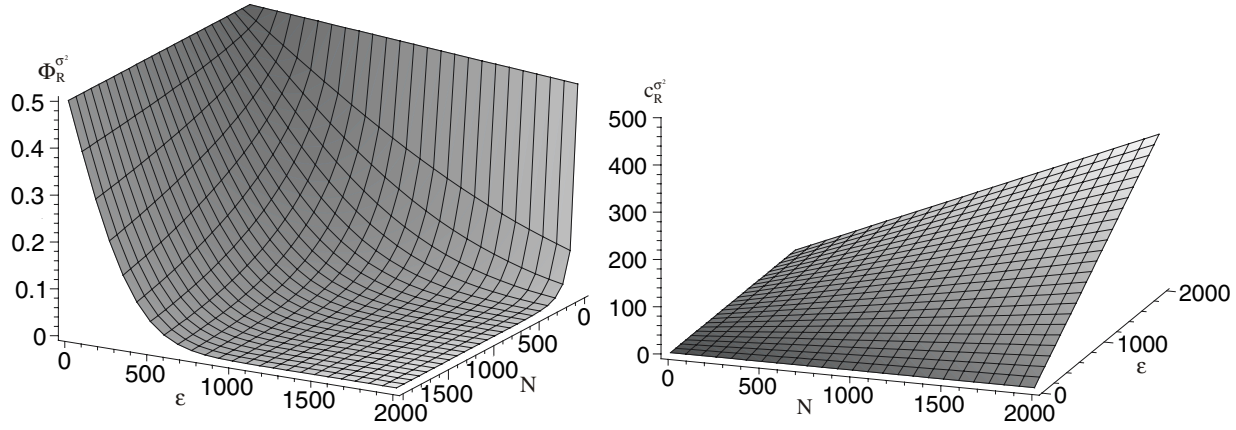


Figure 7: $\Phi_R^{\sigma^2}(N, \varepsilon, \sigma^2)$ **for** $\sigma = 100$ **(left) and** $c_R^{\sigma^2}(N, \varepsilon, \sigma^2, \sigma'^2)$ **for** $\sigma = 100, \sigma' = 150$ **(right).**

5.3 Correlation of data source parameters and packet loss

In this subsection, the correlation of measured parameters of a data source and packet loss is considered. This is shown for the case of a CBR source with silence suppression. The beginning of a talk spurt is normally noticed by the value of the RTP marker bit. If it is set to 1, the current packet belongs to a new talk spurt and the interval between the last packet and this one (minus the duration of the last packet's data) was a silence gap. So, at each arrival of a RTP packet with a set marker bit the IDS is able to update its current measures $\hat{\mu}_S$ and $\hat{\mu}_G$ of the average spurt and gap lengths. Since it is possible that a packet with a set marker bit is lost, those measures are affected by packet loss and this effect has to be taken into account by the IDS. Therefore, the following situation is considered: a CBR source with silence suppression sends packets according to its parameters (an average spurt length of μ_S and an average gap length of μ_G) through a network with a uniformly distributed packet loss rate $l \in [0,1]$. In the following, the expected value $E[\hat{\mu}_S]$ of the measured average spurt length is analyzed.

Let $(\hat{s}_t, \hat{g}_t)$ be the updated measure of the average spurt and gap length resp. after the arrival of the *t*-th packet p_t with a set marker bit. If no packet with a set marker bit is lost between p_{t-1} and p_t the expected value of $\hat{s}_t$ is given by

$$E_0[\hat{s}_t] = \mu_S.$$

If always exactly one packet with a set marker bit gets lost the expected value changes to

$E_1[\hat{s}_t] = \mu_S + \mu_G + \mu_S$.

This can be generalized to the case that exactly *i* packets with a set marker bit get lost:

$E_i[\hat{s}_t] = i(\mu_S + \mu_G) + \mu_S$.

In order to be able to determine the expected measured spurt length occurring at a loss rate l the probability p_i that exactly *i* consecutive packets with a set marker bit are lost is needed. It evaluates to the probability that at first *i* packets are lost and the next packet with a marker bit makes its way through the network, so

$p_i = l^i(1-l)$.

Hence, the requested expected value $E[\hat{\mu}_S]$ is given by

$$E[\hat{\mu}_S] = \sum_{i=0}^{\infty} p_i E_i[\hat{s}_t] = \sum_{i=0}^{\infty} l^i(1-l)(i(\mu_S + \mu_G) + \mu_S)$$

$$= (1-l)\left[(\mu_S + \mu_G)\sum_{i=0}^{\infty} i l^i + \mu_S \sum_{i=0}^{\infty} l^i\right] = (1-l)\left[(\mu_S + \mu_G)\frac{l}{(1-l)^2} + \mu_S \frac{1}{1-l}\right]$$

$$= \frac{\mu_S + l\mu_G}{1-l}.$$

From the graph of $E[\hat{\mu}_S]$, shown in Fig. 8 for $\mu_G = 800$, it can be seen that the mentioned effect is not negligible since e.g. in the case of $\mu_S = 1000$ and $l = 0.1$ the expected measured value is *1200* which is outside of the above-mentioned limit $\varepsilon = 150$.

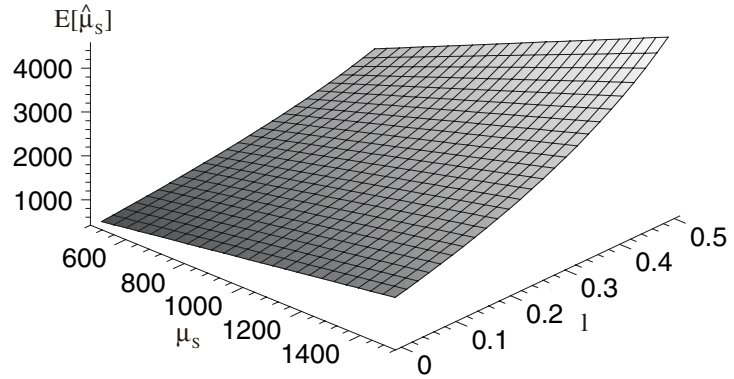


Figure 8: $E[\hat{\mu}_S]$ **for** $\mu_G = 800$**.**

6 Conclusion and Outlook

In this paper we have introduced attacks against data streams generated by multimedia applications and against protocols designed to guarantee QoS. Most of them can only be detected by analyzing statistical values like mean and variance of the object's behavior. We have shown, how the objects for this data communication can be modeled and how the attacks can be detected by using these models. The introduced methods have been evaluated by their false alarm rate and the duration until an on-going attack is detected.

Our future work will focus on the one hand on the test and improvement of the proposed methods using real multimedia streams and on the other hand on an efficient implementation within our ANIDA prototype [BNB01, Büs01]. One aspect will be a possible distribution of the detecting components in order to be able to detect attacks in real-time.

References

[BNB01] R. Büschkes, T. Noll, M. Borning. *Transaction-Based Anomaly Detetion in Communication Networks.* In Proceedings of the 9th International Conference on Telecommunication Systems, Modelling and Analysis, March 2001.

[Bra69] P.T. Brady. *A model for generating on-off speech patterns in twoway conversation.* Bell system Technical Journal, 48(9):2445-2472, September 1969.

[Büs01] R. Büschkes. *Angriffserkennung in Kommunikationsnetzen.* PhD thesis, Aachen University of Technology, May 2001.

[Gol99] D. Gollmann. *Computer Security.* John Wiley & Son, 1999.

[JS00a] W. Jiang and H. Schulzrinne. *Analysis of On-Off Patterns in VoIP and Their Effect on Voice Traffic Aggregation.* In Proceedings of the 9th International Conference on Computer Communications and Networks, October 2000.

[JS00b] W. Jiang and H. Schulzrinne. *Modeling of Packet Loss and Delay and Their Effect on Real-Time Multimedia Service Quality.* In Proceedings of the 10th International Workshop on Network an Operating System Support for Digital Audio and Video, June 2000.

[KBS98] T. Kostas, M. Borella, I. Sidhu, G. Schuster, J. Grabiec, and J. Mahler. *Real Time Voice Over Packet-Switched Networks.* IEEE Network, 12(1):18-27, January/February 1998.

[Ko96] C.C.W. Ko. *Execution Monitoring of Security-Critical Programs in a Distributed System: A Specification-Based Approach.* PhD thesis, University of California, 1996.

[RFC1363] C. Partridge. *A Proposed Flow Specification.* RFC 1363, September 1992.

[RFC1633] R. Braden, D. Clark, and S. Shenker. *Integrated Services in the Internet Architecture: an Overview*. RFC 1633, Juni 1994.

[RFC1889] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson. *A Transport Protocol for Real-Time Applications.* RFC 1889, January 1996.

[RFC2210] J. Wroclawski. *The Use of RSVP with IETF Integrated Services*. RFC 2210, September 1997.

[RFC 2401] S. Kent and R. Atkinson. *Security Architecture for the Internet Protocol.* RFC 2401, November 1998.

[RFC2475] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss. *An Architecture for Differentiated Services.* RFC 2475, December 1998.

[RFC2597] J. Heinanen, F. Baker, W. Weiss, and J. Wroclawski. *Assured Forwarding PHB Group.* RFC 2597, June 1999.

[SBS99] R. Sekar, T. Bowen, and M. Segal. *On preventing intrusions by process behavior monitoring.* In Proceedings of the Workshop on Intrusion Detection and Network Monitoring (ID'99). USENIX, April 1999.

[Tan96] A. S. Tanenbaum. *Computer Networks.* Prentice Hall Press, 1996.

This page has been deliberately left blank

Page intentionnellement blanche

Click here to view PowerPoint presentation; Press Esc to exit

An Analysis of the Kerberos Authentication System

Ian Downard and Dr. Ann Miller
University of Missouri – Rolla
Department of Electrical and Computer Engineering
1870 Miner Circle
116 Emerson Elec Co Hall
Rolla, MO 65409, USA

itd@umr.edu, millera@ece.umr.edu

Key words: computer security, software testing

Abstract

The first of this project's two objectives was to review the current level of trustworthiness inherent to MIT's implementation of the Kerberos authentication standard. The second objective was to consider how various probable abuses could be detected by host-based or network-based Intrusion Detection Systems and demonstrate that Kerberos produces enough auditable information to make intrusion detection a feasible and effective means of defense.

Our vulnerability assessments of Kerberos predominantly used black box testing techniques and information from published security advisories. The most significant unresolved vulnerabilities we found in MIT's implementation include password guessing attacks and (previously unpublished) Denial of Service (DoS) attacks invoked by disabling disk access required for caching tickets. We also addressed issues associated with the trustworthiness of two external subsystems (time synchronization and domain name services) which Kerberos depends on in very fundamental ways.

1 Kerberos Introduction

Kerberos (RFC1510) is a security system developed at the Massachusetts Institute of Technology in 1988 for providing a secure means of communication between incorporated programs and for providing a secure means of authenticating users into a network of incorporated workstations and servers [1][2]. Since secure authentication is the most important achievement of the Kerberos protocol, it is important to know exactly what kind of authentication Kerberos offers. Traditionally, authentication has focused only on verifying the identities of human users (as clients) on a network, while the authenticity of servers has been presumed valid. Kerberos extends the reach of guaranteed identity by also authenticating servers in the Kerberos network (a feature called *mutual authentication*). This assures clients that the services they are requesting will be processed by authorized servers, and not by other machines possibly masquerading as legitimate servers. Furthermore, Kerberos does not ever need to transmit passwords, even in encrypted form, to perform authentication.

After users have been authenticated, most UNIX based Kerberos implementations also offer the following features:

- *remote logins* by way of the "kerberized" `rlogin`, `telnet`, and `ftp` utilities.
- *encrypted communications* with the DES or 3DES encryption algorithm
- *single sign-on capabilities*, so a user can access kerberized services without logging in more than once

The Kerberos protocol illustrated in Figure 1 starts when user, Alice, logs into a client workstation and requests a service from application server, Bob. Alice obtains permission for that service with a service-granting ticket issued from the Ticket-Granting Service (TGS) on the Kerberos server, but first Alice must prove her identity to and obtain a Ticket Granting Ticket (TGT) from the Authentication Service (AS) on the Kerberos server. The TGT gives a user permission to request service tickets which grant access to

Paper presented at the RTO IST Symposium on "Real Time Intrusion Detection", held in Estoril, Portugal from 27-28 May 2002, and published in RTO-MP-101.

application servers. The messages shown in Figure 1 are described in more detail below[1]:

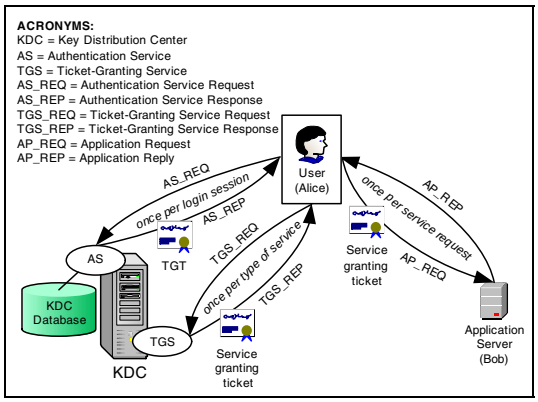


Figure 1 - Kerberos Overview

AS_REQ:

- Alice requests a TGT from the AS
- message format (sent in the clear if preauthentication is not used):
  `UserID: Alice`

AS_REP:

- The AS verifies with the Key Distribution Center (KDC) database that Alice is authorized to request tickets, then (pending authorization) the AS generates and sends to Alice a session key (S_{Alice}) encrypted with Alice's secret key (K_{Alice}), and the TGT.
- message format:
  $K_{Alice}\{$`use` S_{Alice} `with TGS`$\}$,
  `TGT:` $K_{TGS}\{$`use` S_{Alice} `with Alice`$\}$
  where
  $K_{Alice}\{X\}$ is the encryption of X with Alice's secret key,
  S_{Alice} is the *session key* shared between the KDC and Alice, and
  $K_{TGS}\{X\}$ is the encryption of X with the TGS's secret key.

TGS_REQ:

- After decrypting AS_REP and recovering S_{Alice}, Alice requests a service-granting ticket from the TGS, with her authenticator (which proves her identity) and her TGT.
- message format:
  `"Alice is requesting Bob",`
  `Authenticator:` $S_{Alice}\{$`Alice, time`$_1\}$,
  `TGT:` $K_{TGS}\{$`use` S_{Alice} `with Alice`$\}$

TGS_REP:

- After verifying Alice's authenticator, the KDC sends the service-granting ticket to Alice in a message encrypted with S_{Alice}.
- message format:
  $S_{Alice}\{$`use` S_{AB} `with Bob`$\}$,
  `service ticket:` $K_{Bob}\{$`use` S_{AB} `with Alice`$\}$
  where
  S_{AB} is the *session key* shared between Alice and Bob, and
  K_{Bob} is Bob's secret key.

AP_REQ:

- Once Alice recovers S_{AB} and the service ticket from TGS_REP, she encrypts her userID with S_{AB} to prove her identity to Bob, and sends that authenticator along with the service ticket to Bob. Optionally, she can set a flag to turn on mutual authentication so the server will subsequently prove it's identity to her.
- message format:
  $S_{AB}\{$`Alice, time`$_2\}$,
  `service ticket:` $K_{Bob}\{$`use` S_{AB} `with Alice`$\}$
  `mutual authentication flag: on/off`

AP_REP:

- If Alice requested mutual authentication, then Bob extracts `time`$_2$ from AP_REQ and returns it to the client encrypted using S_{AB}.
- message format:
  $S_{AB}\{$`time`$_2\}$

Subsequent messages between Alice and Bob depend on the type of service being used. After initial authentication, some applications proceed to communicate with clear unencrypted messages, while others utilize encrypted sessions.

2 Testing Techniques

It is very important to understand that our testing results are derived from testing MIT's Kerberos in its default configuration. Many, if not all, of the vulnerabilities documented in this report can be avoided by customizing the default Kerberos configuration or by avoiding certain kerberized applications. It is also very important to understand that we have tested MIT's implementation of the Kerberos standard, and that our results only apply to that implementation and not to the standard, or any other implementation.

Some tests in this project were directed at the Kerberos login utility, `kinit`, which obtains and caches an initial ticket-granting ticket (TGT) and session key from the Kerberos server (or more specifically, the KDC). By default, the TGT is valid for ten hours and is saved as `/tmp/krb5cc_[uid]`, where `[uid]` is the

[1] We follow the convention used by Microsoft in [11] for denoting message contents.

user's identification number. Users must hold a valid TGT before they can request service tickets from the Kerberos server. Service tickets are necessary for users to gain access to other services, like mail or ftp. Requests for service tickets are transparent to users who already hold valid TGTs.

Thus TGTs are fundamental to Kerberos functionality, and exploits which attack `kinit` can compromise the entire Kerberos system.

2.1 Password Guessing with `kinit`

A Perl script was written to demonstrate how MIT's Kerberos in its default configuration does not implement protection against online password guessing attacks [3]. This script helped characterize the behavior of Kerberos against multiple invalid authentication attempts given via the `kinit` login utility. Output consisted of the combined time it took `kinit` to prompt the user for a password and then accept or deny authentication based on the user's input[2]. This output was plotted on a time per guess set of axes, and is included in Figure 2. During this experiment, passwords were guessed at an average rate of 120.42 passwords per second.

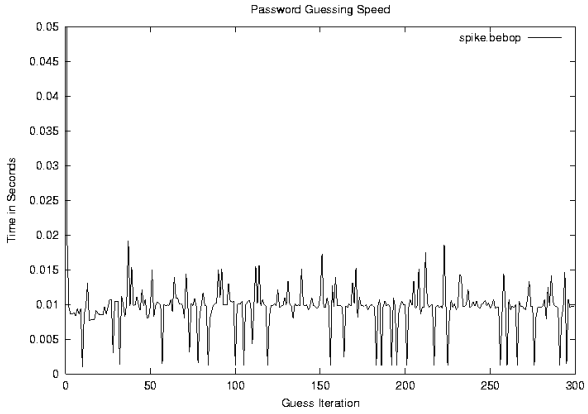


Figure 2: Password guessing speed for computer "spike.bebop"

Two common techniques used to guard against password guessing attacks include increasing the response time to login attempts, and temporarily invalidating a user's access to the system after some number of sequential invalid login attempts. The data shown in Figure 2 illustrates that MIT's Kerberos doesn't implement either of these techniques, since the vast majority of the times remain relatively close to the mean time, and the user's account was never disabled even after 300 invalid login attempts.

Implementing password policies with the `kadmin` utility and limiting the rate of ticket requests any single source may ask from a Kerberos server [3] are two methods which can greatly limit the effectiveness of online password guessing attacks.

Kerberos is also vulnerable to *offline* password guessing attacks because an attacker can identify Alice's secret key by repeatedly guessing it until one can decrypt the encrypted session key in the AS_REP message. This vulnerability can be limited by requiring *preauthentication* before issueing a TGT [4].

2.2 Vulnerable ticket cache[3]

Since Kerberos saves tickets (or "credentials cache") in the `/tmp` area, which is globally writable, a denial of service attack could be implemented by disabling write access to the `/tmp` directory. This can be achieved by any unprivileged user and was demonstrated on our Kerberos realm by exhausting the `/tmp` partition with large files. When `kinit` is unable to perform a write to this area, Kerberos fails to authenticate the user. `Kinit` produced the error, "`Credentials cache I/O operation failed XXX when initializing cache`" after attempting to authenticate a user without having enough temporary disc space to write ticket cache data.

Another method to deny `kinit` write access to the `/tmp` area can be done in the following three steps:

1. Predict a user's ticket cache filename (for example, `/tmp/krb5cc_[uid]`, where `[uid]` is the decimal user identification number of a legitimate Kerberos user).
2. Create a file named the same as the ticket cache filename, with, `touch /tmp/krb5cc_[uid]`.
3. Change the permissions on the file to be read only with, `chmod 444 /tmp/krb5cc_[uid]`.

When the victimized user tries to login into the Kerberos system with `kinit`, it will deny them access and return the error, "`kinit: Internal file credentials cache error when initializing cache`".

Denying kerberized authentication utilities like `kinit` write access to the `/tmp` area could either refuse users login access altogether, or

[2] The time needed for the user to enter a password was excluded from this measurement.

[3] This vulnerability is heretofore unpublished.

they could default to some less secure, but functionally equivalent program. If, for example, the kerberized `rlogin` fails, then some systems will revert to using the legacy `rlogin` executable, which transfers all the user's activity in clear and unencrypted text over a potentially insecure network where a malicious sniffer could procure sensitive information like legitimate usernames and passwords.

This vulnerability was demonstrated on releases 1.1.1 and 1.2.1 of MIT's Kerberos 5 for Linux, but it's severity can be limited if users always specify alternate locations for their credentials cache in areas that are more robust than `/tmp` and less predictable than `/tmp/krb5cc_[uid]`[4]. Additional security can be gained if administrators remove all the legacy applications for which there is a kerberized replacement (`telnet`, `ftp`, `rlogin`, `rsh`, etc.).

2.3 Insecure Subsystems

Kerberos relies on but does not authenticate some external services inherent to a computer's operating system or network. Since a system is only as secure as its weakest link, any vulnerabilities in these subsystems will be inherited by the systems that depend on them. In this section, we analyze the vulnerabilities Kerberos inherits from time synchronization services and the domain name system.

2.3.1 Time services

By default, Kerberos requires that all hosts and servers have clocks accurate within 5 minutes of each other. If a host's clock exceeds this tolerance, then Kerberos denies access to all users and services from that host. This clock skew tolerance is an approximation designed to minimize the effectiveness of replay attacks where intruders may reuse intercepted network traffic to obtain illegitimate authentication and/or continue to use expired tickets. Thus, time services, such as the Network Information Service (NIS) and the Network Time Protocol (NTP), can directly impact the reliability of Kerberos. Assuming trustworthy time services isn't always a valid assumption[5], as is the case for the popular NIS [5]. The vulnerabilities Kerberos could inherit from subverted time services include:

- Replay attacks of valid tickets and/or reuse of expired tickets [3].
- DoS attacks issued by moving a server's clock beyond the clock skew tolerance.

2.3.2 Domain Name System

Virtually every system residing in the OSI application layer relies on the Domain Name System (DNS) [6] to resolve hostnames to IP addresses. MIT's Kerberos trusts DNS for locating KDCs, client workstations, and application servers. Failures in the DNS standard almost always fall into the following two categories:

1. incorrect mapping between IP addresses and symbolic hostnames (e.g. spoofed entries)
2. unanswered resolution queries (e.g. denial of service)

These failures can potentially disable all Kerberos facilities.

However, most DNS related problems originate from system dependant DNS implementations, such as BIND, whose vulnerabilities range from the mundane to the catastrophic (like unauthorized root access). As long as the DNS server is physically isolated from each Kerberos server in a realm, then a catastrophic failure in DNS would still be limited to DoS on Kerberos. Mutual authentication helps prevent DNS failures from becoming a larger problem, possibly involving client communication with a masquerading server.

This issue was described by Hornstein and Altman in [7] as a problem with the Kerberos trust model, rather than the Kerberos protocol. As such, the only practical solution involves replacing DNS in favor of another more trustworthy name resolution service, like DNSSEC (DNS with security extensions).

3 Recent Security Advisories

The Computer Emergency Response Team Coordination Center (CERT/CC) and MIT have published a number of recent advisories regarding Kerberos 5. Table 1 lists some of the more urgent issues [8][5].

Kerberos 5 supports backwards compatibility with Kerberos 4. Consequently, when defects are discovered in Kerberos 4, they often manifest themselves in Kerberos 5 as well. MIT has advised the Kerberos community about two such vulnerabilities, which include DoS

[4] This can be achieved by using the KRB5CCNAME environment variable.

[5] The problem of insecure or unreliable time services is one that affects almost every distributed computer system.

attacks on the KDC, and buffer overruns which can be exploited for unauthorized root access.

Table 1: Recent security advisories in MIT's implementation of Kerberos 5

Advisory	Date	Description	Category of attack
CERT Advisory CA-2000-11	Jun 9, 2001	Buffer overflow vulnerabilities expose the KDC to various DoS attacks. The vulnerabilities do not appear to permit an attacker to execute code.	DoS
CERT Advisory CA-2000-06	May 17, 2001	Buffer overflow vulnerabilities exist in `krshd`, `ksu`, and two library functions used by many Kerberos authenticated services.	Unauthorized root access
Telnetd buffer overflows	Jul 31, 2001	A buffer overflow vulnerability exists in the telnet daemon, which may give attackers unauthorized root access.	Unauthorized root access
Ftpd buffer overflows	Apr 25, 2001	An attacker can gain unauthorized root access via the FTP daemon if anonymous FTP is enabled or the attacker has access to a local account on the affected system.	Unauthorized root access
Unsafe temp file handling in krb4 code	Mar 7, 2001	Systems running login daemons with Kerberos 4 support are vulnerable to an exploitable race condition when the login daemons running as root are creating new ticket files in `/tmp`. This vulnerability makes it possible for an attacker to overwrite arbitrary files as root.	Unauthorized modification of files
Remote Root vulnerability in GSSFTP program	Jun 14, 2001	A single misnamed variable in the FTP daemon source code effectively permits any user to execute certain FTP commands without authorization.	DoS, and unauthorized root access

All of the security advisories mentioned in this section have been resolved in several recent updates to MIT's Kerberos, but analyzing where past vulnerabilities have occurred helps identify areas where security was not built-in from the early stages of development. For example, four of the six advisories in Table 1 originate from buffer overflow problems, so it is logical for future testing to dedicate extra effort analyzing modules which deal with memory or I/O management. Furthermore, attackers can be identified by Intrusion Detection Systems which alert on activity that resembles *any* known attack signature, regardless of whether the vulnerability has been resolved.

4 Auditing and Intrusion Detection Capabilities

Symptoms of intrusions and abuses in a Kerberos system may appear in client application output, syslog output, Kerberos logs, and KDC generated network traffic. Therefore, both network-based and host-based Intrusion Detection Systems (IDSs) can be useful methods for detecting intrusions.

4.1 Host-Based Intrusion Detection

All the information Kerberos servers consider useful for auditing can be logged to a configurable destination, such as the `/var/log/krb5kdc.log` file, and Kerberos client applications generally log all their auditable information via syslog and `/var/log/messages`. These files are the best resources available for host-based intrusion detection because they provide a significant data base from which keyword matching, pattern matching, and user profiling algorithms can be applied. Appendix A shows an example KDC log file which contains the information we used for detecting suspect activity with simple keyword matching. We emphasized intrusion detection with the KDC log file instead of syslog output because it more accurately represents the usage of Kerberos services.

Keyword: `Request is a replay`

It is possible for an attacker to receive an unauthorized service from an application server by intercepting and replaying a legitimate service request from another user. Any instance of this warning on a reliable network (in terms of data errors) should raise a high priority alarm since it would likely indicate a replay attack. However, on unreliable networks, this warning may simply be a byproduct of using UDP, so a high priority alarm may not be appropriate.

Keyword: `CLIENT_NOT_FOUND`

The KDC logs this error whenever a client requests authentication for a user who is not

registered in the Kerberos database. Many repeated instances of this warning should raise a high priority alarm since they may indicate attempts by attackers to guess qualified usernames. Seldom or occasional instances of this warning should raise no alarms since they would probably be the result of honest user mistakes.

Keyword: `UNKNOWN_SERVER`

Mistyped server names and FTP requests are the two most common causes for this error. FTP requests cause this error because of a benign contradiction between the Secure FTP RFC and a design decision in Kerberos to not make any ftp/<hostname> principles [9]. Seldom or occasional instances of this warning should raise no alarms.

Keyword: `Clock skew too great`

Any client machine or application server whose clock differs from the Kerberos server by more than 5 minutes (or some other configured clock skew threshold) cannot participate in the Kerberos network. Therefore, if an attacker can alter a host machine's clock, then they can effectively deny all users and services on that host access to Kerberos authentication facilities. If time synchronization services (like the Network Time Protocol, NTP) are running between all the hosts and Kerberos servers in a network, then clock skew errors should raise a high priority alarm because it would indicate a failure in that time service, and a denial of service to the effected host.

If time synchronization services are not in place, then hosts with unattended clocks will eventually drift beyond the skew threshold. Clock skew errors would most likely indicate that consequence. In this case, a low priority alarm should suffice to alert its administrator of his neglected duty to monitor that host's clock.

Keyword: `ISSUE`

The "ISSUE" keyword appears whenever the KDC *issues* a TGT or TGS in response to an AS_REQ or TGS_REQ message (respectively). These actions are logged together with the affiliated user and server identifiers (called *principles*). Host-based IDSs can use this information to build user profiles for use in statistical anomaly detection. These algorithms are used to determine if observed behavior is normal, so that changes in a user's behavior can be closely scrutinized. An example of suspect changes in user behavior is shown in Figure 3.

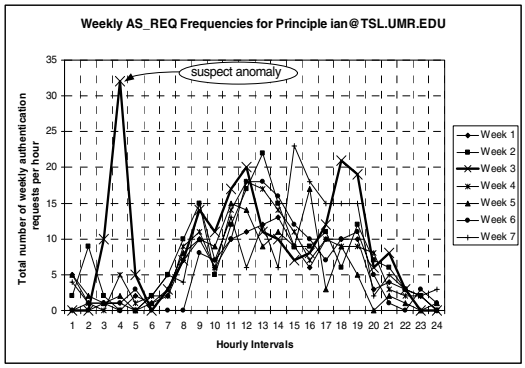


Figure 3 - Weekly frequency of AS_REQ messages from user, ian@TSL.UMR.EDU.

Figure 3 shows the hourly frequency of authentication requests issued per week by a single user. The peak frequency which occurs during the 4th hour of week 3 is an example of the kind of aberrant activity which should be classified as suspicious since that sudden increase in authentication requests does not resemble the user's average behavior.

The KDC log entries regarding AS_REQ and TGS_REQ provide enough information to profile the following login and session characteristics of users:

- login frequency by day, time, and location
  - This can detect when intruders log into a user's account from hosts which are rarely accessed or at times when the account is usually inactive.
- frequency, type and location from where services are requested
  - This can detect when intruders request services in a manner not characteristic of the compromised user account.

Keyword: `Ticket expired`

This error indicates some user's TGS request failed because that user's TGT had expired. This error would not likely indicate an intrusion, so it should not raise any alarms, but if it's immediately followed by an authentication request, then it can be used to help build user profiles.

Finally, any massively repeated warning or error in the KDC log file may indicate attempts by an attacker to overload the KDC, or exhaust resources required for logging facilities. This situation should always raise a high priority alarm.

Thus, vigilantly monitoring the KDC log file is an essential requirement to any host-based IDS

for the Kerberos environment. Utilities which can monitor system log files include:

- Tripwire (http://tripwire.com),
- Swatch (http://oit.ucsb.edu/~eta/swatch), and
- Logcheck (http://psionic.com/abacus/logcheck).

4.2 Network-Based Intrusion Detection

In addition to reporting errors to a log file, KDCs also return error descriptions to clients whose ticket requests have failed. The defined type for these messages is KRB_ERROR, and they contain error codes which clients can translate to users in human readable form. By intercepting these messages a Network-based Intrusion Detection System (NIDS) can help protect a Kerberos network from attempted attacks. We applied this idea by writing rules for an open source NIDS, called Snort [10], which would alert when and report why various authentication requests failed. Appendix B contains a complete listing of these rules, and the following paragraphs summarize them.

4.2.1 Snort Overview

Snort is an open source NIDS created by Marty Roesch and valued by many in the security field as a very high quality security tool for UNIX based environments because of its excellent documentation, efficiency, and configurability. It applies signature analysis techniques for detecting intrusions and provides a facility for users to expand the signature set with their own rules.

4.2.2 Snort Rule Definitions

One approach to defining rules applicable to Kerberos generated network traffic is to analyze the error messages transmitted by KDCs. Another approach involves analyzing the traffic generated by Kerberos clients and application servers. However, this second approach will produce rules very specific to a single Kerberos environment because deciding which kerberized applications to support is the subjective prerogative of each system's administrator. Rules based on traffic from KDCs will be more universally applicable to Kerberos environments because the KDC is such an essential component of the infrastructure. While we emphasized the first approach for defining NIDS rules, we also addressed the question of how rules may be defined for specific applications by analyzing the kerberized remote shell (`rshd`) and `ftp` applications.

Our proposed NIDS can detect many of the same problems as the host-based approach because many of the authentication errors which are communicated to clients are also logged in the KDC log files.

Some intrusions which affect the Kerberos system are more likely to be detected than others, so we also extended Snort's list of categories and priorities for classifying alerts with the entries listed in Table 2 in Appendix C. This way, alerts which are often false alarms can be assigned a low priority.

4.2.3 Rules for KDC-to-Client Traffic

The rules mentioned in this section only encompass some of the errors which may be symptoms of an intrusion. Many more rules can be defined which detect intermittent failures in the Kerberos protocol that are less obviously related to actual intrusions. We chose to include some of these extra rules in our Snort rule set listed in Appendix B, but they would be inappropriate for networks where limiting false alarms is a high priority.

`KRB5KDC_ERR_C_PRINCIPAL_UNKNOWN`

This error occurs whenever a client has tried to authenticate with a user identifier (called a *principal*) which has not been registered in the Kerberos database. Occasional instances of this error are probably benign, but excessively repeated instances may indicate a username or password guessing attack.

`KRB5KRB_AP_ERR_REPEAT`

This error indicates that a service or authentication request has been replayed, and it should warn the client's administrator that an attacker may be sniffing packets on the network and attempting to gain illegitimate authentication.

`KRB5KRB_AP_ERR_SKEW`

This error advises clients that their system clock has exceeded the maximum allowable clock skew. Administrators for those hosts must adjust their clocks back within the clock skew threshold before Kerberos services can be resumed to that host. This error may indicate an attack on a system's time services.

If UDP is the transport protocol being used, the following four errors should be extremely unusual, and if TCP is being used, they should never occur. Therefore, an administrator should

consider that an attacker is modifying messages whenever one of these errors occurs.

- **KRB5KRB_AP_ERR_BAD_INTEGRITY**
  This error occurs when a message cannot be decrypted due to lost integrity.
- **KRB5KRB_AP_ERR_MODIFIED**
  This error occurs when the message stream has been modified (i.e. lost integrity).
- **KRB5KRB_AP_ERR_BADSEQ**
  This error occurs when a message is received out of sequence.
- **KRB5KRB_AP_ERR_INAPP_CKSUM**
  This error occurs when a message's checksum fails to verify the message's integrity.

4.2.4 Rules for Kerberized Applications

Three rules were defined to illustrate how a NIDS might address the buffer overflow vulnerabilities associated with `rshd` (see the CERT Advisory CA-2000-06 listed in Table 1). Various messages which contain the string "too long" will be detected by these rules and alerted as a possible buffer overflow attempt. Another rule was defined to detect failed authentication attempts with `rshd` by alerting when messages contained the pattern "Permission denied".

One last rule was defined for detecting attempts at exploiting the remote root vulnerability in the GSSFTP program (see Table 1). This is rule alerts on messages which contain the pattern "user root", which is often a prohibited command even for users who know the root password.

5 Open Questions

The results which have been presented here are applicable to MIT's implementation of the Kerberos protocol, and do not necessarily apply to other distributions. Testing other distributions could possibly identify vulnerabilities which are commonly found across multiple Kerberos implementations, such as those from Cisco Systems, CyberSafe, DCE Security, IBM, Microsoft [11], and Sun Microsystems.

While our ideas for detecting attacks on Kerberos were experimentally proven on a closed network with synthesized intrusions and simplified IDSs, the realistic utility of these ideas is largely theoretical. Future work should apply the intrusion detection rules and anomaly detection concepts presented here in a real-world, attack prone network where issues like false alarms and slow performance are important.

6 Conclusions

Two major points of vulnerability in the Kerberos authentication system observed during this project were password guessing attacks and denial of service (DoS) attacks. Denying Kerberos write access to disk space used for storing ticket cache data was determined to be the most significant DoS vulnerability due to the potential consequences and ease of abuse. The consequences of this attack range from refusing a user login access to causing unencrypted passwords to travel across insecure networks. Monitoring disk usage in areas where user credentials are stored, like `/tmp/`, can reliably detect attacks based on this vulnerability.

The password guessing vulnerability [3][4] is of concern because password guessing algorithms can quickly guess poorly formulated passwords. These attacks can be detected from highly repeated authentication requests and resisted by enforcing password policies and/or using preauthentication.

MIT's Kerberos also inherits DoS and replay vulnerabilities through its fundamental dependence on unauthenticated time services [3] and DNS name resolution services [7]. Kerberos depends on reliable time services for expiring old tickets and will fail to serve a host whose clock skew is beyond a specified threshold. DNS is relied upon for addressing Key Distribution Centers (KDCs), host workstations, and servers in a network. The vulnerabilities inherited by Kerberos from DNS depend on whether the DNS servers are isolated from the Kerberos servers. If they are not isolated, then Kerberos may be subject to all the consequences of unauthorized root access. If they are isolated, then the vulnerabilities are limited to DoS attacks.

Intrusion Detection Systems (IDSs) can help identify when an attacker is trying to exploit these and other Kerberos vulnerabilities, such as replay attacks, and message integrity attacks. One advantage to using a network-based IDS is that they can be placed anywhere in the Kerberos network, whereas the host-based IDS would need to be placed on the machine where KDC logs are kept. The disadvantage of the network based approach, is that if the IDS is not running on the KDC, then it must be running on every Kerberos client machine. Assuming the IDS is itself immune to intrusion, the best approach to Kerberos intrusion detection would place both a host-based and network-based IDS on the Kerberos server running the KDC.

We described how a host-based IDS could detect password guessing attacks, replay attacks, DoS attacks which target time synchronization

facilities, and other intrusions which don't resemble normal system activities. The KDC logs may contain symptoms of these attacks and provide enough data to build user profiles for statistical anomaly detection.

We used KDC-to-client network traffic to define rules for a network-based IDS, called Snort [10], which successfully alerted on password guessing attacks, replay attacks, and various attacks on message integrity. Other Snort rules were defined for the kerberized remote shell (`rshd`) and `ftp` utilities to alert on buffer overflow attacks and unauthorized attempts to gain administrator privileges (respectively).

Acknowledgements

This research was made possible by support through the Graduate Assistance in Areas of National Need (GAANN) fellowship. This work was partially funded by the NSF Grant DUE-0113949. Nathan Neulinger of University of Missouri – Rolla provided valuable insight and suggestions.

References

[1] Massachusetts Institute of Technology. *Kerberos: The Network Authentication Protocol.* 2000: http://web.mit.edu/kerberos/www/.

[2] J. Kohl, *The Kerberos Network Authentication Service (V5)*. C. Neuman, Editor. 1993: http://www.ietf.org/rfc/rfc1510.txt

[3] S.M. Bellovin, "Limitations of the Kerberos Protocol," *Winter 1991 USENIX Conference Proceedings*, USENIX Association, pp. 253-267.

[4] J. Pato. Using Pre-authentication to Avoid Password Guessing Attacks. DCE-RFC 26.0. June 1993: http://www.opengroup.org/tech/rfc/mirror-rfc/rfc26.0.txt.

[5] CERT Coordination Center. 2001: http://www.cert.org/.

[6] P. Mockapetris, Domain Names – Implementation and Specification. Nov 1987: http://www.ietf.org/rfc/rfc1035.txt.

[7] Hornstein, Altman, *Distributing Kerberos KDC and Realm Information with DNS*, August 28, 2001 (expiration): http://www.ietf.org/internet-drafts/draft-ietf-krb-wg-krb-dns-locate-02.txt

[8] Massachusetts Institute of Technology. *Kerberos Security Advisories*. 2001: http://web.mit.edu/kerberos/www/advisories/.

[9] NCSA Kerberos Troubleshooting for UNIX. 2001: http://archive.ncsa.uiuc.edu/General/CC/kerberos/troubleshooting.html

[10] Snort by Martin Roesch. 2002: http://www.snort.org/.

[11] Microsoft TechNet. *Windows 2000 Kerberos Authentication*. 1999: http://www.microsoft.com/technet/prodtechnol/windows2000serv/deploy/confeat/kerberos.asp

About the Authors

Ian Downard is a graduate student at the University of Missouri-Rolla (UMR) working towards a master's degree in Computer Engineering. He received a BS in Computer Engineering at UMR in 2000. As a graduate student, Ian has taught a course on computer science algorithms, has held internships at the Naval Research Laboratory, and is presently researching computer security topics.

Ann Miller is the Cynthia Tang Missouri Distinguished Professor of Computer Engineering at UMR. Prior to this, she was the Deputy Assistant Secretary of the United States Navy for Command, Control, Communications, Computing, Intelligence, Electronic Warfare, and Space. She has also held senior engineering positions in industry.

Appendix A

This appendix contains a sample KDC log file.

```
Mar 29 15:46:43 feistel.tsl.umr.edu krb5kdc[11325](info): TGS_REQ 130.100.1.33(88):
PROCESS_TGS: authtime 0, <unknown client> for host/feistel.tsl.umr.edu@TSL.UMR.EDU,
Request is a replay

Mar 29 19:11:50 feistel.tsl.umr.edu krb5kdc[11325](info): AS_REQ 130.100.1.33(88):
CLIENT_NOT_FOUND: fred@TSL.UMR.EDU for krbtgt/TSL.UMR.EDU@TSL.UMR.EDU, Client not found
in Kerberos database

Mar 29 19:11:50 feistel.tsl.umr.edu krb5kdc[11325](info): DISPATCH: repeated
(retransmitted?) request from 130.100.1.33 port 88, resending previous response

Mar 30 13:15:15 feistel.tsl.umr.edu krb5kdc[11325](info): TGS_REQ 130.100.1.20(88):
UNKNOWN_SERVER: authtime 1017602107, wilma@TSL.UMR.EDU for
ftp/laptop02.tsl.umr.edu@TSL.UMR.EDU, Server not found in Kerberos database

Mar 30 15:41:27 feistel.tsl.umr.edu krb5kdc[11112](info): TGS_REQ 130.100.1.33(88):
PROCESS_TGS: authtime 0, <unknown client> for host/feistel.tsl.umr.edu@TSL.UMR.EDU, Clock
skew too great

Mar 31 23:24:54 feistel.tsl.umr.edu krb5kdc[10856](info): TGS_REQ 130.100.2.14(88):
PROCESS_TGS: authtime 0, <unknown client> for host/feistel.tsl.umr.edu@TSL.UMR.EDU,
Ticket expired

Mar 31 15:29:43 feistel.tsl.umr.edu krb5kdc[11112](info): AS_REQ 130.100.1.33(88): ISSUE:
authtime 1017437383, barny@TSL.UMR.EDU for krbtgt/TSL.UMR.EDU@TSL.UMR.EDU

Mar 31 15:29:46 feistel.tsl.umr.edu krb5kdc[11112](info): AS_REQ 130.100.1.33(88): ISSUE:
authtime 1017437386, deno@TSL.UMR.EDU for krbtgt/TSL.UMR.EDU@TSL.UMR.EDU

Mar 31 15:29:49 feistel.tsl.umr.edu krb5kdc[11112](info): TGS_REQ 130.100.1.33(88):
ISSUE: authtime 1017437386, pebbles@TSL.UMR.EDU for host/feistel.tsl.umr.edu@TSL.UMR.EDU
```

Appendix B

This appendix contains a list of rules for the Snort network intrusion detection system to alert possible abuses of Kerberos services. See the Snort user's manual [10] for the rules description language.

```
#############################################################################
# These are rules for monitoring kerberos related traffic into a Kerberos
# client machine.
# The ' content: "|02 01 1e|" ' rule denotes that this message is a kerberos
# message type of KRB_ERROR.  The other content rules identify specific rules
# documented in the Kerberos V5 system administrator's guide.  The 02 01
# prefix in these rules was observed with ethereal to precede every error code.
# Also, note that the error codes in the sysadmin manual are given in decimal,
# while their value in KRB5_ERROR packets are specified in hex.  Therefore, our
# rules must match their hex values.
#
# The snort users manual defines the syntax for user-defined rules like these.
#############################################################################

var MY_IP 130.100.1.33/32
var KRB_SRC_PORT 88
var KRB_CLIENT_PORTS any
var KSHELL_SRC_PORT 544
var FTP_SERVER_PORT 21

# KRB5KDC_ERR_C_PRINCIPAL_UNKNOWN: Client not found in Kerberos database
alert udp any $KRB_SRC_PORT -> $MY_IP $KRB_CLIENT_PORTS (msg:"Unknown client principle"; content: "|02 01 06|"; \
 content: "|02 01 1e|"; nocase; classtype:failed_client_authentication;)
alert tcp any $KRB_SRC_PORT -> $MY_IP $KRB_CLIENT_PORTS (msg:"Unknown client principle"; content: "|02 01 06|"; \
 content: "|02 01 1e|"; nocase; classtype:failed_client_authentication;)
    # test with: kinit -p bogusname

# KRB5KRB_AP_ERR_BAD_INTEGRITY: Decrypt integrity check failed
alert udp any $KRB_SRC_PORT -> $MY_IP $KRB_CLIENT_PORTS (msg:"Decrypt integrity check failed"; content: "|02 01 1f|"; \
 content: "|02 01 1e|"; nocase; classtype:mesg_integrity;)
alert tcp any $KRB_SRC_PORT -> $MY_IP $KRB_CLIENT_PORTS (msg:"Decrypt integrity check failed"; content: "|02 01 1f|"; \
 content: "|02 01 1e|"; nocase; classtype:mesg_integrity; priority:1;)

# KRB5KRB_AP_ERR_REPEAT: Request is a replay
alert udp any $KRB_SRC_PORT -> $MY_IP $KRB_CLIENT_PORTS (msg:"Message replay detected"; content: "|02 01 22|"; \
 content: "|02 01 1e|"; nocase; classtype:mesg_replay;)
alert tcp any $KRB_SRC_PORT -> $MY_IP $KRB_CLIENT_PORTS (msg:"Message replay detected"; content: "|02 01 22|"; \
 content: "|02 01 1e|"; nocase; classtype:mesg_replay;)
    # tcpreplay -l 1 /root/packets/telnet02

# KRB5KRB_AP_ERR_SKEW: Clock skew too great
alert udp any $KRB_SRC_PORT -> $MY_IP $KRB_CLIENT_PORTS (msg:"Clock skew threshold exceeded"; content: "|02 01 25|"; \
 content: "|02 01 1e|"; nocase; classtype:successful_dos;)
alert tcp any $KRB_SRC_PORT -> $MY_IP $KRB_CLIENT_PORTS (msg:"Clock skew threshold exceeded"; content: "|02 01 25|"; \
 content: "|02 01 1e|"; nocase; classtype:successful_dos;)
    # test by changing client's clock by a couple hours, and issueing a TGS_REQ (AS_REQ seems to not care about clock
    # skews, but TGS will be rejected)

# KRB5KRB_AP_ERR_BADADDR: Incorrect net address
alert udp any $KRB_SRC_PORT -> $MY_IP $KRB_CLIENT_PORTS (msg:"Incorrect net address"; content: "|02 01 26|"; content: \
 "|02 01 1e|"; nocase; classtype:unknown_net_addr;)
alert tcp any $KRB_SRC_PORT -> $MY_IP $KRB_CLIENT_PORTS (msg:"Incorrect net address"; content: "|02 01 26|"; content: \
 "|02 01 1e|"; nocase; classtype:unknown_net_addr;)

# KRB5KRB_AP_ERR_MODIFIED: Message stream modified
alert udp any $KRB_SRC_PORT -> $MY_IP $KRB_CLIENT_PORTS (msg:"Message integrity lost"; content: "|02 01 29|"; \
 content: "|02 01 1e|"; nocase; classtype:mesg_integrity;)
alert tcp any $KRB_SRC_PORT -> $MY_IP $KRB_CLIENT_PORTS (msg:"Message integrity lost"; content: "|02 01 29|"; \
 content: "|02 01 1e|"; nocase; classtype:mesg_integrity; priority:1;)

# KRB5KRB_AP_ERR_MUT_FAIL: Mutual authentication failed
alert udp any $KRB_SRC_PORT -> $MY_IP $KRB_CLIENT_PORTS (msg:"Mutual authentication failed"; content: "|02 01 2e|"; \
 content: "|02 01 1e|"; nocase; classtype:failed_server_authentication;)
alert tcp any $KRB_SRC_PORT -> $MY_IP $KRB_CLIENT_PORTS (msg:"Mutual authentication failed"; content: "|02 01 2e|"; \
 content: "|02 01 1e|"; nocase; classtype:failed_server_authentication;)

# KRB5KRB_AP_ERR_BADSEQ: Incorrect sequence number in message
alert udp any $KRB_SRC_PORT -> $MY_IP $KRB_CLIENT_PORTS (msg:"Message out of sequence"; content: "|02 01 31|"; \
 content: "|02 01 1e|"; nocase; classtype:mesg_integrity;)
alert tcp any $KRB_SRC_PORT -> $MY_IP $KRB_CLIENT_PORTS (msg:"Message out of sequence"; content: "|02 01 31|"; \
 content: "|02 01 1e|"; nocase; classtype:mesg_integrity; priority:1;)

# KRB5KRB_AP_ERR_INAPP_CKSUM: Inappropriate type of checksum in message
alert udp any $KRB_SRC_PORT -> $MY_IP $KRB_CLIENT_PORTS (msg:"Invalid checksum"; content: "|02 01 32|"; content: "|02 \
 01 1e|"; nocase; classtype:mesg_integrity;)
alert tcp any $KRB_SRC_PORT -> $MY_IP $KRB_CLIENT_PORTS (msg:"Invalid checksum"; content: "|02 01 32|"; content: "|02 \
 01 1e|"; nocase; classtype:mesg_integrity; priority:1;)

#############################################################################
# The following rules attempt to alert intrusions issued by way of the
# kerberized remote shell server (see the rshd[8] man page for a
# list of error diagnostics returned by rshd through network sockets)
#############################################################################

# This generates an error because usernames must be less than 16 characters.
alert tcp any $KSHELL_SRC_PORT -> $MY_IP any (msg: "kshd\: Local username greater than 16 characters"; content: \
 "locuser too long"; classtype:buffer_overflow;)
    #test with: rsh feistel -x -l 12345678123456789 "echo helloworld"
```

```
alert tcp any $KSHELL_SRC_PORT -> $MY_IP any (msg: "kshd\: Remote username greater than 16 characters"; content: \
 "remuser too long"; classtype:buffer_overflow;)
alert tcp any $KSHELL_SRC_PORT -> $MY_IP any (msg: "kshd\: Command arguement list is too long"; content: "command too \
 long"; classtype:buffer_overflow;)
alert tcp any $KSHELL_SRC_PORT -> $MY_IP any (msg: "kshd\: Authentication procedure failed"; content: "Permission \
 denied"; classtype:suspicious_login;)

# The following rule attempts to alert misuse of the local ftp service
alert tcp any any -> $MY_IP $FTP_SERVER_PORT (msg: "FTP root login attempted"; content: "user root"; nocase; \
 classtype: attempted-admin;)
    #this won't work for encrypted ftp sessions
```

Appendix C

This appendix contains a list the categories used to classify the Kerberos rules in Appendix B. The entries in this table correspond to entries in Snort's `classification.config` file.

Table 2 - Alert classifications, where lower priority numbers indicate more urgent the alerts.

Class Name	Description	Priority
failed_client_authentication	suspicious principle used for authentication	2
failed_server_authentication	server failed to mutually authenticate to client	2
mesg_integrity	possible message corruption by an interceptor	3
mesg_replay	previous message has been captured and replayed	1
unknown_net_addr	invalid net addr may indicate DNS problems	3
buffer_overflow	buffer overflow attack may have been attempted with client application	3
suspicious_login	attempted login using a suspicious username	3
successful_dos	possible denial of service attack has occurred	2
attempted-admin	attempted administrator privilege gain	1

This page has been deliberately left blank

Page intentionnellement blanche

Click here to view PowerPoint presentation; Press Esc to exit

Intrusion Detection Systems for IP Telephony Networks

Martin Steinebach
Fraunhofer Institute IPSI
Dolivostr.15
64293 Darmstadt
Germany

martin.steinebach@ipsi.fraunhofer.de

Frank Siebenhaar, Christian Neubauer
Fraunhofer Institute IIS
Am Weichselgarten 3
91058 Erlangen
Germany

sbn@iis.fhg.de, neu@iis.fhg.de

Jana Dittmann
Fraunhofer Institute IPSI-C4M
Competence for Media Security
Dolivostrasse, 15
D-64293 Darmstadt
Germany

jana.dittmann@platanista.de

Utz Roedig, Ralf Ackermann
KOM Multimedia Communiucations
Merckstr. 25
64283 Darmstadt
Germany

Utz.Roedig@KOM.tu-darmstadt.de
Ralf.Ackermann@KOM.tu-darmstadt.de

Summary
Intrusion detection systems (IDS) provide security for network systems. They are used in computer networks to detect violations against security policies or unusual events that could lead towards a security thread. Telephone networks based on the internet protocol (IP) called IP telephony (IPT) are a recent development in network usage and will become a common application in the next years as they can provide integrated services based on telephony communications.
We identify IPT security demands as well as risks and analyze the possibility of adding IDS concepts to IPT systems. Digital audio watermarking is a technology able to provide different security aspects in this context. The combination of classic IDS and audio watermarking leads to new and promising way of user and data authentication and monitoring of calls.

1. Background and Motivation
In this section we briefly introduce the IP telephony scenario and the technology used for IP telephony Furthermore we discuss the security risks of IP technology inclusion.

1.1 Pre-IP telephony networks
Telecommunication is often an area with a high security demand. The Internet on the other hand is a risky and insecure environment. Therefore methods for securing IPT are necessary to ensure user acceptance of this new service. For existing telephony systems fraud detection is an established service. In the IPT network, similar services are required and IDS often functions to protocol connections and to discriminate between correct use and fraud.
Figure 1 shows a telecommunication network including IP-telephony terminals (T) and standard telephones (image). Telephony security is based on the identification of the telephone used for communication. “Identification by wire” provides a connection between the possibility of using the network and a correct identification of calling and called party. A manipulation can only take place in the physical domain. Another telephone can be attached at the telephone wire to use the same caller ID as the original owner of the number.

1.2 IP Telephony
IPT scenario networks are more complex due to the connection to the classical telecommunication networks. Computers or terminals (which are basically simple computers) are used instead of telephones. Requirements are a multimedia capability for recording and playback of audio data and IP access. Different protocols like H.323 or SIP can be applied to achieve connectivity with other calling

Paper presented at the RTO IST Symposium on “Real Time Intrusion Detection”, held in Estoril, Portugal from 27-28 May 2002, and published in RTO-MP-101.

partners. Audio codecs are used for AD/DA conversion and compression of the audio data resulting in a trade-off between sound quality and bandwidth requirements.

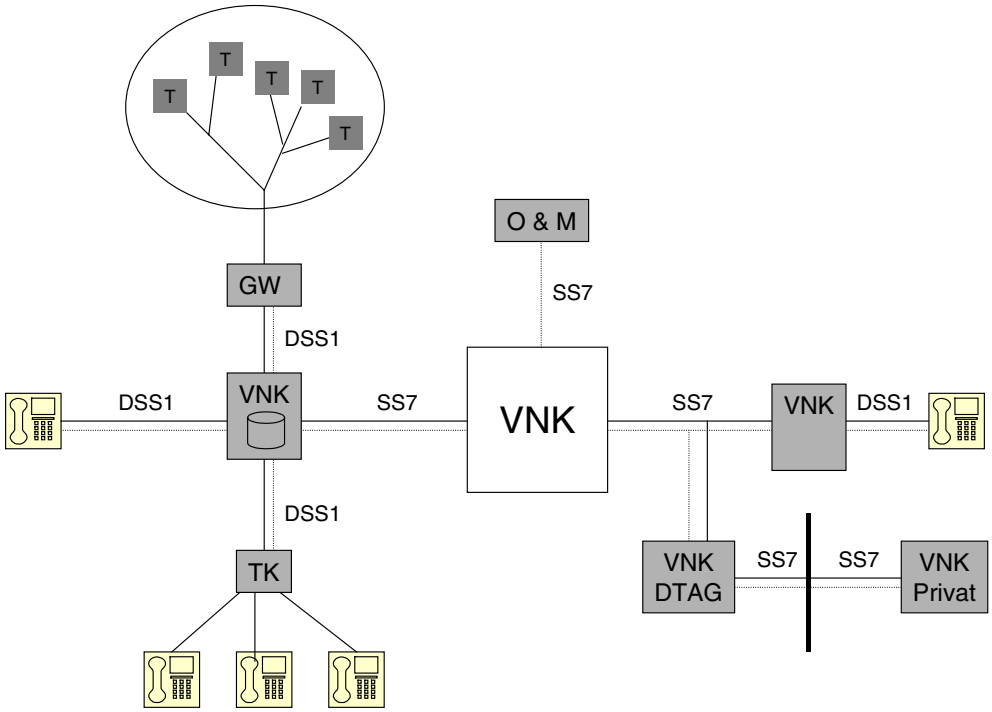


Figure 1: Mixed telecommunication network

1.3 IP-telephony security issues

By using a computer for communication, especially trough a network, identification by wire is lost. Instead a identification by computer or by user ID can take place. Both can be attacked as they are based on transmitted data. Computer networks offer numerous points of attack. Tools for spoofing can be used to act as another calling party. In [ASRS01] security issues are discussed in more detail.
Intrusion detection is a widely accepted technology to address security problems in computer networks (see e.g. [SBD+91]). But, compared to most computer networks secured by an intrusion detection system (IDS), a telephony network is much more complex:

- IP telephony switches from computer to telephony networks and/or vice versa
- network connections can be international, using multinational servers for connection
- mobility is an important issue for IP telephony, a caller or terminal may move from one network to another, taking rights and policies with him / it.

Therefore new, distributed concepts are necessary to achieve IPT intrusion detection. In our paper we introduce an approach by combining IDS concepts with traditional telecommunication based methods. Furthermore we integrate digital watermarking into our IDS-concept. In an IPT network, spoken words are transmitted as sampled audio data. Digital watermarking technologies can provide solutions for authenticity and integrity risks. Digital watermarks can be transparently added to existing voice over IP systems and are designed to survive the applied compression codecs.

2. Digital watermarking

Digital Watermarking is a powerful technology capable of solving important practical security problems like authentication for copyright protection. Watermarking techniques usually used for digital imagery and now also used for audio and 3D-models are relatively new and are growing at an exponential rate. Well over 90% of all publications in this field have been published in the last 7 years. It is a highly multidisciplinary field that combines image and signal processing with cryptography, communication theory, coding theory, signal compression, and the theory of visual perception. Interest in this field has recently increased because of the wide spectrum of applications it addresses.

As an introduction, we discuss the main watermarking parameters. The most important properties of digital watermarking techniques are robustness, security, imperceptibility/ transparency, complexity, capacity and possibility of verification [Dit00], [CMB2002].

- **Robustness** describes if the watermark can be reliably detected after media operations. We emphasize that robustness does not include attacks on the embedding scheme that are based on the knowledge of the embedding algorithm or on the availability of the detector function. Robustness means resistance to "blind", non-targeted modifications, or common media operations.
- **Security** describes if the embedded watermarking information cannot be removed beyond reliable detection by targeted attacks based on a full knowledge of the embedding algorithm and the detector, except the key, and the knowledge of at least one watermarked data. The security aspect also addresses the false positive detection rates.
- **Transparency** is based on the properties of the human visual system or the human auditory system. A transparent watermark causes no artifacts or quality loss.
- **Complexity** describes the effort and computational time we need for watermark embedding and retrieval. This parameter is essential if we have real-time applications. Another aspect addresses if we need the original data in the retrieval process or not. Here we distinguish also between non-blind and blind watermarking schemes which influences the complexity.
- **Capacity** describes how many information bits can be embedded. It addresses also the possibility of embedding multiple watermarks in one media in parallel.
- The **verification** procedure describes if we have a private verification like private key functions or a public verification possibility like the public key algorithms in cryptography.

The optimization of the parameters is mutually competitive if we want to embed a large message, we cannot require at the same time large robustness. A reasonable compromise is always a necessity.
We proposed a media independent classification scheme as a basis for quality evaluation in [Dit00]. It is oriented on the application areas where watermarking techniques can be used to meet the needs of the users. Usually existing watermarking techniques can be used in several applications but in each application it is hard to fulfill all quality demands. We find the following watermarking classes based on application areas for digital watermarking:

- **Authentication or Copyright Watermark:** Ensuring copyright protection by watermarking the data with an owner or producer identification
- **Fingerprint Watermark:** Ensuring copyright protection by watermarking the data with customer identifications to track and trace legal or illegal copies
- **Copy Control or Broadcast Watermark:** Ensuring copyrights with customer rights protocols, for example for copy or receipt control
- **Annotation Watermark:** Ensuring copyrights by annotations or capturing of the media data, this kind of watermark is also used to embed descriptions of the value or content of the data
- **Integrity Watermark:** beside the authentication of the author or producer we want to ensure integrity of the date and recognize manipulations

For the IP-telephony intrusion detection scenario, the following applications are of special interest:

- **Authentication watermarking** can ensure the correct identity of calling parties or terminals. A watermark embedded into the voice stream will be harder to attack then a authentication sent by a network protocol. Both calling parties and the IDS can use the embedded watermark to identify the source of the audio stream. This requires a framework for key exchange like a public key infrastructure. Figure 2 shows an example scenario. Here A could use his secret key to encrypt his ID information before it is embedded. Now B who knows who claims to have called him can receive the public key of A and try to decode the embedded information. If this is successful, B can be sure that A has encoded the ID. Current time and data e.g. would be a good candidate for a embedded message, as this would disable replay attacks.
- **Integrity watermarking** can ensure the audio stream has not been tampered. A mechanism in the called terminal and/or in the IDS sensors can identify if a fragile watermark embedded in the calling terminal has been destroyed. Various concepts for fragile watermarking exist, resulting in different possible concepts for integrity protection in this scenario ranging from detection of time gaps to content protection of spoken information.

- **Broadcast watermarking** could be used to identify attackers listening to communications. If a watermark with the ID of the called party is embedded into the audio stream, a IDS sensor could identify this ID and check if the called party can be found in the network the IDS is monitoring. If this is not the case, the IDS can inform a firewall to block the incoming audio stream and prevent any user inside the network from receiving the audio data,

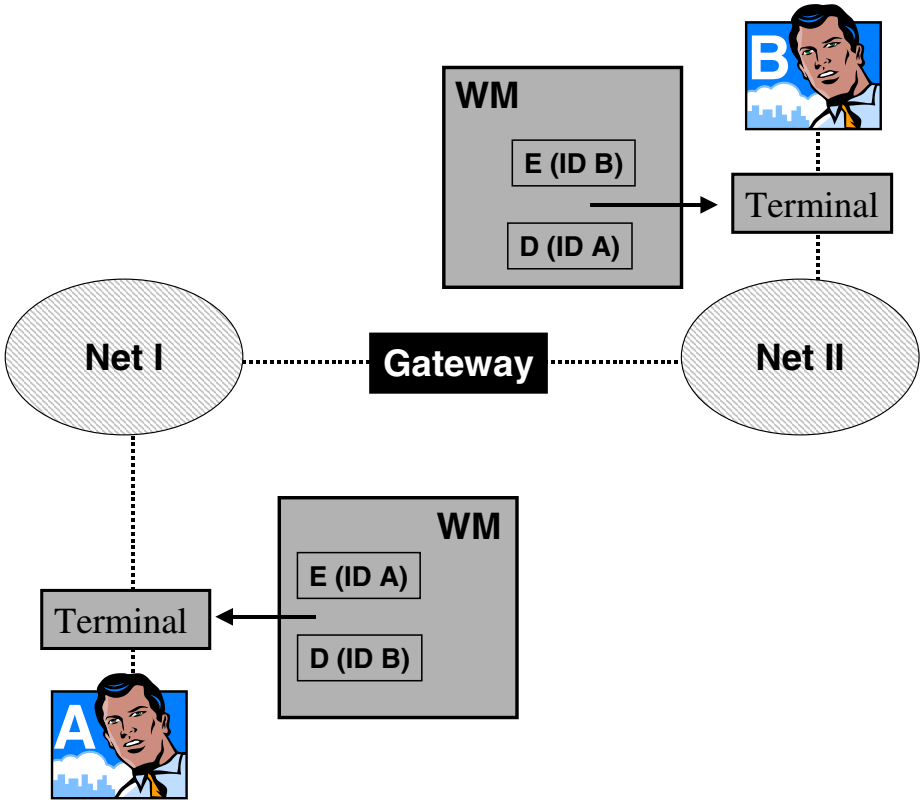


Figure 2: A simple watermarking-based authentications scenario.

2.1 Audio watermarking robustness to telephony

To establish a security concept for IP-telephony based on digital audio watermarking, the applied watermarking algorithms have at least to be robust against the inherent attacks of the scenario:

1. **Low bit rate audio compression:** In voice communication, a strong reduction of the amount of transmitted data is usual. The audio data may be present in high quality with 44,1 kHz and 16 bit at the terminal for the embedding algorithm. For transmission, it will be reduced to e.g. 4 bit and 8 kHz.
2. **Codec changes:** In IP telephony, a compression scheme change may occur while transmission. A gate may recognize a data rate not supported in the local network and change the audio format to another compression scheme.
3. **DA/AD conversion**: Not all communication lines in a telephony scenario will be digital. In some cases a change from digital to analogue audio representation will occur.

(1) has been addressed in our paper “Audio watermarking robustness to lossy compression” [REF]. Here we show that even very low bit rates like mp3 with 30 kbps can be survived if the algorithm has been optimized for high robustness. (3) has been the subject of our paper “Audio watermarking robustness to DA/AD conversions”. Like with lossy compression, a high robustness is possible if a loss of perceived audio quality is accepted at the same time. We show that even analogue transmissions via speaker and microphone can be survived. While a simple change from digital to analogue transmission has not been the subject of our tests, first experiments showed a very high robustness to this attack.

Figure 2 shows an example scenario: An authentication watermark for user A is embedded into the PCM voice data. Now a number of attacks against the robustness of the watermark are applied to the audio stream:

- **Lossy compression** of the PCM stream as discussed above.
- The voice stream can be subject to **packet loss** in the network. Usually in real-time environments it is not possible to resend the audio packets. Therefore they are dropped from the stream and result in gaps compared to the original stream. This will be a challenge to the synchronization capabilities of the watermarking algorithm.
- **Format conversion**: The lossy compression format may be changed at a gateway as described above under "codec changes".
- **Packet losses** can also occur in net II.
- The audio data is **converted to PCM** data. Here usually a change of sample rate and number of bits for sample representation will occur, resulting in an addition of noise to the signal.

Now user B tries to identify the calling party by detection of an authentication watermark in the audio stream A watermarking algorithm suited for the scenario has to survive all the attacks above at the same time or is not robust against the scenario. If the watermark has been destroyed, B will not find A's ID and can not trust the communication.

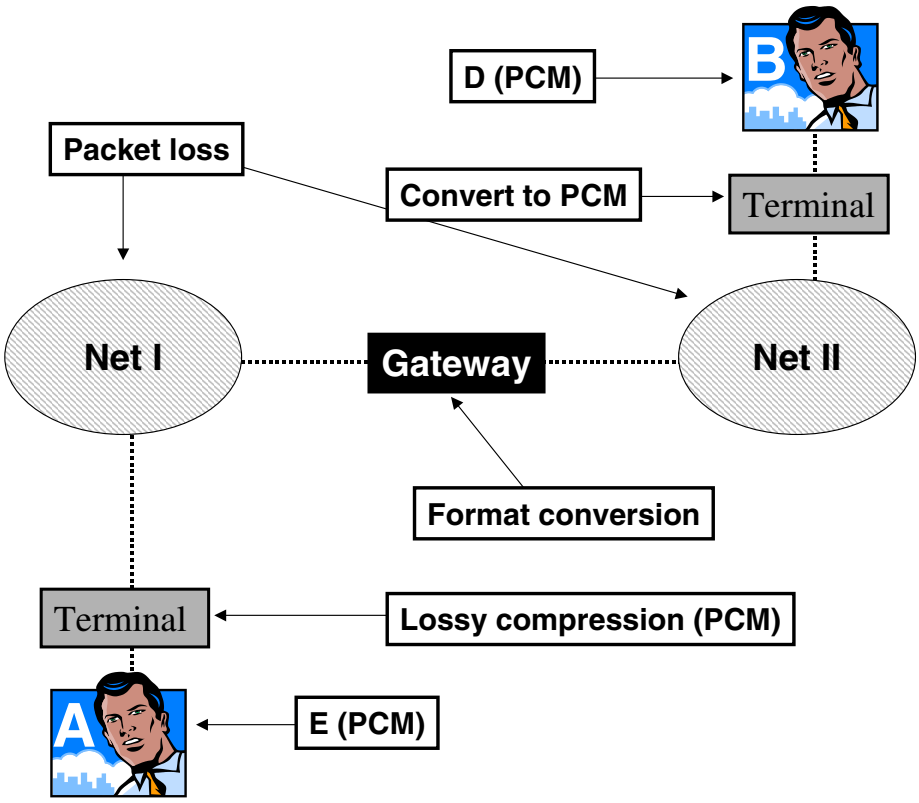


Figure 3: Various attacks against an embedded watermark

2.1.1 Test results

Here we discuss first test results for digital audio watermarking robustness to IP-Telephony based compression schemes. We use our IPSI audio watermarking prototype at different robustness modes and bit rates and checked its robustness against the following compression methods:

- GSM 6.10 at 44,1 and 8 Khz
- mu-Law (CCITT standard G.711)
- Microsoft ACM: Lernout Hauspie CELP 4,8 kBit/s
- 3-bit IMA/DVI ADPCM (5.3:1)
- 4-bit Dialogic ADPCM

The original audio material was encoded in 16 bit, 44,1 kHz , mono. Five different files containing spoken text in English language at various quality levels have been used as examples.

- High payload watermarks with a data rate of ~30 bits per second are robust against GSM 6.10 coding at 44,1 kHz.
- Medium payload watermarks with a data rate of ~7 bits per second are robust against mu-Law, 3-bit IMA/DVI ADPCM and 4-bit Dialogic ADPCM
- Low payload watermarks with a data rate of ~1 bit per second are robust against GSM at 8 kHz.

All watermarking modes have shown to be robust against compression methods higher-payload modes are robust against. The low payload watermark is robust against very high compression rates: GSM at 8 kHz has a compression rate of 1:53 compared to the original audio format.

2.2 Real-time audio watermarking

Besides robustness, low complexity of the watermarking algorithm is a very demanding aspect. By nature, IP-telephony is intended to be a real-time application. This means that the complete processing in the terminals – watermark embedding, audio encoding, and sending of the data via IP in case of the calling party terminal or receiving of the data, audio decoding, and watermark detection in the called party terminal – must be carried out in real-time. Moreover, IP-telephony in general is a two-way communication in which each terminal acts as sender and receiver simultaneously. This intensifies the requirement for low complexity systems.
IP terminals can be computers or also modified telephones which are capable of establishing a connection to the Internet. Considering the first case, the computational complexity for IP-telephony should be low enough that the user can work with other programs at the same time, e.g. a text editor to make some notes about the call. In the latter case, an overall low complexity of all processing steps and, thus, also the watermark embedding/detection algorithm is an important issue in order to minimise the costs for the terminals.

2.2.1 Test results

All prototypic implementations of the watermarking schemes developed at the Fraunhofer IPSI and the Fraunhofer IIS perform several times faster than real-time on a conventional computer system. In the following, the complexity measurement results of the PCM watermarking scheme of the Fraunhofer IIS are presented. This scheme offers a very robust and transparent watermark embedding with a watermark data rate of up to 48 bit/s. The tests were carried out on a state-of-the-art Pentium-4 based PC with 1.6 GHz and 512 MByte of main memory. The complexity numbers given in table 1 are normalised to the duration of the test items (monophonic and stereophonic, 16 bits per sample, 44100 Hz sampling frequency), i.e. 1.0 corresponds to execution of the application in real-time and a number below 1.0 denotes a computation time less than the playing time.

Program	Mono	Stereo
Watermark Embedder	0.070	0.135
Watermark Detector	0.156	0.314

Table 1: Complexity numbers of the Fraunhofer IIS PCM watermarking software evaluated on a Pentium-4 based PC-system.

The table shows that especially in the case of monophonic audio signals – the default signal type for IP-telephony – the watermark embedder performs around 14 times faster and the watermark detector more than six times faster than real-time. However, it has to be kept in mind that this watermarking scheme was developed for high quality audio material. The execution time can be decreased significantly if the parameters of the watermarking scheme are adapted to the needs of an IP-telephony application. As an example, the bandwidth of the input signals for the watermark embedder is usually

limited by a pre-filtering step and, therefore, the calculation of the watermark has only to be performed for this reduced bandwidth.
Thus, it can be stated that the extension of real-time IP-telephony applications with digital watermarking is already possible. Due to the low complexity of the watermarking steps there is still enough remaining computational power for additional processes that can be executed in parallel when applied on modern computer systems. Furthermore, this means that the watermarking functionality can be integrated in specialized hardware terminals using low cost chips.

3 Watermarking based Intrusion Detection

In section 2 we discuss the usability of digital audio watermarking in an IP telephony scenario. As robustness and complexity are both suited for the application, we now describe our concept of watermarking-based intrusion detection for IP telephony as well as other basic security functions based on this approach. In our paper [DKLS2002] we introduce a watermarking-based multimedia firewall. Here a detector is connected with a firewall. Incoming or outgoing media data for which a detector is present is analyzed. If a watermark is successfully detected or (taking the opposite approach, can not be detected properly), different reactions are possible. The most basic one is simply blocking the data. Protocol functions are also described.

3.1 User Authentication

To implement security mechanisms in IP-telephony systems, user authentication is an important requirement. We propose the following concept (see also figure 2):

1. Calling party terminal: Watermark is embedded in audio stream
2. Network: Watermark is transported with the audio data. In case of codec-changes or DA/AD conversion, the watermark stays present.
3. Called party terminal: Watermark is detected

Based on the embedded watermark the called party can now identify the caller. This makes it hard for a third party to spoof a connection as the specific watermarking key is not known. The called party can also detect sudden changes in the embedded identity and e.g. alert the caller or simply block the connection. The security of the embedded watermark can be increased by applying more sophisticated protocol extensions. The embedded caller ID, actual time, called-party ID and other info can be encrypted using a PKI private key and then be retrieved and decrypted using the corresponding public key.

3.2 Data Authentication

Integrity protection can work similarly to the authentication protocol. We have already proposed concepts for audio integrity protection in [DSS2001]. We propose to embed a time code, a content description or both with a robust watermark in the audio stream. At the other end of the connection, the mark is retrieved. Now changes in sequence of audio packages or manipulations of the content can be detected by comparing the audio data with the embedded mark.

3.3 Intrusion Detection System

Like the end systems, an ID-System for IP-Telephony environments is able to use the watermarking information in the audio stream to fulfill its tasks. The resulting advantage for an ID-System using the watermark information in addition (or as a replacement) to the information extracted from the IP-telephony signaling flows is the following.
An IP-Telephony call might be routed through different protocol clouds (e.g. a SIP and a H.323 cloud) using appropriate gateways. To be able to use IDS-Mechanisms that cover both protocol worlds, security related information regarding the call, have to be present and mapped appropriately in both clouds. Therefore, a gateway has to translate security related signaling protocol elements in addition to the signaling elements used for call signaling and call control. Current available gateways do not possess this capability, because a definition for a security related protocol mapping between different IP-Telephony standards is missing. Additionally, if a call is transported via a non IP Telephony based link (such as a conventional telephone line) parts of the signaling information usually gets lost at the terminating gateways and only the audio data and what is embedded within stays available. By using watermark information in the audio stream, the mentioned gap could be covered. Because the

necessary user-information is present in the audio stream this information is present in all protocol-clouds that are involved in a call. This includes calls that are routed between PSTN and IP-Telephones. In such a scenario, an adaptation between both signaling clouds to support security mechanisms would be hard to achieve. Even gateways that perform a transcoding of the audio streams might be used if robust watermarks are used.
An ID-System using the advantage of watermark information needs access to the PK-Infrastructure that is used for the watermarking process of all involved Telephony systems. In addition the ID-System needs the capability to access the audio streams to extract the watermark. Such an ID-System is capable to detect a number of (suspicious) misuse situations. In the following we give a subset of examples that can be used to qualify a call or a set of calls as possible attacks:

1. Calls that are originated or terminated from users that are not permitted to use the service (because it either has no or a false ID for a certain service, e.g. calling abroad).
2. Frequency of calls within a certain time period (e.g. “War dialing”) – those can even be originated from different protocol clouds (both IP Telephony and non IP Telephony)
3. Suspicious call patterns such as a number of parallel calls originated from different locations but using the same unique ID. This might indicate that a certain account has been corrupted.

When comparing the approach with existing security mechanisms (including authentication and protection) it has a number of advantages. The protection scheme and its integration with IDS mechanisms can cover several heterogeneous protocol clouds and call transmission links. The approach can both be used end-to-end as well as only on parts of the (call) link. Systems that are not able to process watermarks do not interfere with systems that are capable to do so.

4 Summary and Conclusion

The audio watermarking algorithms in our tests are robust against compression codecs typically applied in IP-Telephony. With high compression rates, the bit rate of the algorithms has to drop to provide sufficient robustness. At compression rates of above 50:1, we can still embed ~1 bit per second. Therefore we see audio watermarking as a powerful new technology to improve IP-telephony security. We show how audio watermarking and IDS can be combined to solve open security problems in IP Telephony .
One first approach could be user authentication by embedding user Ids into the audio streams. This also leads to the possibility of new intrusion detection methods based on monitoring the embedded IDs. Current challenges are key management for watermark embedding and detection and computational power for watermarking-enabled IDS sensors.
Later concepts could also include embedding of time codes to disable replay attacks based on recorded voice messages and for detecting gaps in the audio stream. Integrity protection be embedding content-fragile watermarks is also on interesting perspective, but the required data rate combined with a high required robustness makes this a future research topic.

5 Future Work

To develop a watermarking-based intrusion detection system for IP-Telephony and similar applications, a number of improvements are necessary in audio watermarking and framework domain.
Payload and reliability regarding errors have to be improved if one requires integrity protection besides authentication. Our current research shows that content integrity protection for audio watermarking is only possible by applying checksums for feature descriptors.
Our multimedia firewall prototype based on watermark detectors can be seen as a proof of concept. Future research has to improve processing speed of the firewalls’ watermark detector to reduce delay and required buffer memory.
At the Transmark project we are working on improvements in real-time audio and video watermark embedding and detection. A lower computational cost for both embedding and retrieval enables a broader analysis of media streams as a single IDS will be capable of monitoring more connections. At the same time, the requirements for the terminals embedding the watermark in real-time will be reduced.
To summarize our future activities, we will improve watermarking technology and integrate it into existing frameworks for new media-based security systems.

Acknowledgments

Parts of this research have been funded by T-Nova and the BMBF project TransMark.

Literature

[ASRS01] Ralf Ackermann, Markus Schumacher, Utz Roedig, Ralf Steinmetz: Vulnerabilities and Security Limitations of current IP Telephony Systems; Proceedings of Communication and Multmedia Security 2001, Kluver, 2001

[CMB2002] I. Cox, M. Miller, J. Bloom, Digital Watermarking, 2002, Academic Press, San Diego, USA, ISBN 1-55860-714-5

[Ditt00] Dittmann, Jana: Digitale Wasserzeichen, Springer Verlag, ISBN 3 - 540 - 66661 - 3, 2000

[DKLS2002] Jana Dittmann, Stephan Klink, Andreas Lang, Martin Steinebach, "Wasserzeichenunterstützende Firewalls", Enterprise Security: Grundlagen, Strategien, Anwendungen, Realisierungen, Patrick Horster (Hrsg.), it Verlag für Informationstechnik GmbH, Höhenkirchen, pp. 246 - 257, ISBN 3-936052-03-4,2002.

[DSS2001] Dittmann, Jana; Steinebach, Martin; Steinmetz, Ralf (2001). Merkmale digitaler Audiodaten zur Verwendung in inhaltsfragilen digitalen Wasserzeichen. In: Verlässliche IT-Systeme 2001, Sicherheit in komplexen IT-Infrastrukturen, Vieweg & Sohn Verlagsgesellschaft mbH, Braunschweig/Wiesbaden, pp. 193 - 208, ISBN 3-528-05782-3, 2001..

[RAS01] Utz Roedig, Ralf Ackermann, Ralf Steinmetz. Sicherheit von IP- Telephonie-Szenarien, Februar 2001.

[RATWS00] Utz Roedig, Ralf Ackermann, Marc Tresse, Lars Wolf, Ralf Steinmetz. Verbesserte Systemsicherheit durch Kombination von IDS und Firewall. In Systemsicherheit, March 2000

[SBD+91] S. Snapp, J. Brentano, G. Dias, T. Goan, T. Grance, et al.; A System for Distributed Instrusion Detection, Proc. of the 14th Department of Energy Computer Security Group Conference, May 1991, pp.(17)25-(17)45.

This page has been deliberately left blank

Page intentionnellement blanche

REPORT DOCUMENTATION PAGE

1. Recipient's Reference	2. Originator's References	3. Further Reference	4. Security Classification of Document
	RTO-MP-101 AC/323(IST-033)TP/18	ISBN 92-837-0032-5	UNCLASSIFIED/ UNLIMITED

5. Originator Research and Technology Organisation
North Atlantic Treaty Organisation
BP 25, F-92201 Neuilly-sur-Seine Cedex, France

6. Title
Real Time Intrusion Detection

7. Presented at/sponsored by
the RTO Information Systems Technology Panel (IST) Symposium held in Estoril, Portugal, 27-28 May 2002.

8. Author(s)/Editor(s)	9. Date
Multiple	June 2003
10. Author's/Editor's Address	**11. Pages**
Multiple	236 (text) 634 (slides)

12. Distribution Statement There are no restrictions on the distribution of this document. Information about the availability of this and other RTO unclassified publications is given on the back cover.

13. Keywords/Descriptors

Communications management
Communications networks
Computer networks
Computer security
Correlation techniques
Data fusion
Data processing security
Decision support
Denial of service attacks
Electronic security
IDS (Intrusion Detection Systems)
Information systems
Information warfare
Intrusion detectors
Monitors
Network security
Network traffic
Protocols
Real time operations
Reporting
Requirements
Secure communication
Software engineering
Surveillance
Threat evaluation
Traffic simulation
Visualization technologies
Vulnerability

14. Abstract

This volume contains the Technical Evaluation Report, the Keynote Addresses and 17 papers, presented at the Information Systems Technology Panel Symposium held in Estoril, Portugal from 27th to 28th May 2002.

The papers presented covered the following headings:

- Real-Time Intrusion Detection, Overview and Practical Experience
- Correlation and Fusion
- Insider Threat Detection
- Real-Time Data Analysis and Processing
- Real-Time Decision Support and Visualisation
- Intrusion Detection for Real-Time and Time-Service Dependent Applications

This page has been deliberately left blank

Page intentionnellement blanche

NORTH ATLANTIC TREATY ORGANISATION

RESEARCH AND TECHNOLOGY ORGANISATION
BP 25 • 7 RUE ANCELLE
F-92201 NEUILLY-SUR-SEINE CEDEX • FRANCE
Télécopie 0(1)55.61.22.99 • E-mail mailbox@rta.nato.int

DIFFUSION DES PUBLICATIONS
RTO NON CLASSIFIEES

L'Organisation pour la recherche et la technologie de l'OTAN (RTO), détient un stock limité de certaines de ses publications récentes, ainsi que de celles de l'ancien AGARD (Groupe consultatif pour la recherche et les réalisations aérospatiales de l'OTAN). Celles-ci pourront éventuellement être obtenues sous forme de copie papier. Pour de plus amples renseignements concernant l'achat de ces ouvrages, adressez-vous par lettre ou par télécopie à l'adresse indiquée ci-dessus. Veuillez ne pas téléphoner.

Des exemplaires supplémentaires peuvent parfois être obtenus auprès des centres nationaux de distribution indiqués ci-dessous. Si vous souhaitez recevoir toutes les publications de la RTO, ou simplement celles qui concernent certains Panels, vous pouvez demander d'être inclus sur la liste d'envoi de l'un de ces centres.

Les publications de la RTO et de l'AGARD sont en vente auprès des agences de vente indiquées ci-dessous, sous forme de photocopie ou de microfiche. Certains originaux peuvent également être obtenus auprès de CASI.

CENTRES DE DIFFUSION NATIONAUX

ALLEMAGNE
Streitkräfteamt / Abteilung III
Fachinformationszentrum der Bundeswehr, (FIZBw)
Friedrich-Ebert-Allee 34
D-53113 Bonn

BELGIQUE
Etat-Major de la Défense
Département d'Etat-Major Stratégie
ACOS-STRAT-STE – Coord. RTO
Quartier Reine Elisabeth
Rue d'Evère, B-1140 Bruxelles

CANADA
DSIGRD2
Bibliothécaire des ressources du savoir
R et D pour la défense Canada
Ministère de la Défense nationale
305, rue Rideau, 9e étage
Ottawa, Ontario K1A 0K2

DANEMARK
Danish Defence Research Establishment
Ryvangs Allé 1, P.O. Box 2715
DK-2100 Copenhagen Ø

ESPAGNE
INTA (RTO/AGARD Publications)
Carretera de Torrejón a Ajalvir, Pk.4
28850 Torrejón de Ardoz - Madrid

ETATS-UNIS
NASA Center for AeroSpace Information (CASI)
Parkway Center
7121 Standard Drive
Hanover, MD 21076-1320

FRANCE
O.N.E.R.A. (ISP)
29, Avenue de la Division Leclerc
BP 72, 92322 Châtillon Cedex

GRECE (Correspondant)
Defence Industry & Research General Directorate
Research Directorate
Fakinos Base Camp
S.T.G. 1020
Holargos, Athens

HONGRIE
Department for Scientific Analysis
Institute of Military Technology
Ministry of Defence
H-1525 Budapest P O Box 26

ISLANDE
Director of Aviation
c/o Flugrad
Reykjavik

ITALIE
Centro di Documentazione Tecnico-Scientifica della Difesa
Via XX Settembre 123a
00187 Roma

LUXEMBOURG
Voir Belgique

NORVEGE
Norwegian Defence Research Establishment
Attn: Biblioteket
P.O. Box 25, NO-2007 Kjeller

PAYS-BAS
Royal Netherlands Military Academy Library
P.O. Box 90.002
4800 PA Breda

POLOGNE
Armament Policy Department
218 Niepodleglosci Av.
00-911 Warsaw

PORTUGAL
Estado Maior da Força Aérea
SDFA - Centro de Documentação
Alfragide
P-2720 Amadora

REPUBLIQUE TCHEQUE
DIC Czech Republic-NATO RTO
VTÚL a PVO Praha
Mladoboleslavská ul.
Praha 9, 197 06, Česká republika

ROYAUME-UNI
Dstl Knowledge Services
Kentigern House, Room 2246
65 Brown Street
Glasgow G2 8EX

TURQUIE
Millî Savunma Başkanliği (MSB)
ARGE Dairesi Başkanliği (MSB)
06650 Bakanliklar - Ankara

AGENCES DE VENTE

NASA Center for AeroSpace Information (CASI)
Parkway Center
7121 Standard Drive
Hanover, MD 21076-1320
Etats-Unis

The British Library Document Supply Centre
Boston Spa, Wetherby
West Yorkshire LS23 7BQ
Royaume-Uni

Canada Institute for Scientific and Technical Information (CISTI)
National Research Council
Acquisitions
Montreal Road, Building M-55
Ottawa K1A 0S2, Canada

Les demandes de documents RTO ou AGARD doivent comporter la dénomination "RTO" ou "AGARD" selon le cas, suivie du numéro de série (par exemple AGARD-AG-315). Des informations analogues, telles que le titre et la date de publication sont souhaitables. Des références bibliographiques complètes ainsi que des résumés des publications RTO et AGARD figurent dans les journaux suivants:

Scientific and Technical Aerospace Reports (STAR)
STAR peut être consulté en ligne au localisateur de ressources uniformes (URL) suivant:
http://www.sti.nasa.gov/Pubs/star/Star.html
STAR est édité par CASI dans le cadre du programme NASA d'information scientifique et technique (STI)
STI Program Office, MS 157A
NASA Langley Research Center
Hampton, Virginia 23681-0001
Etats-Unis

Government Reports Announcements & Index (GRA&I)
publié par le National Technical Information Service
Springfield
Virginia 2216
Etats-Unis
(accessible également en mode interactif dans la base de données bibliographiques en ligne du NTIS, et sur CD-ROM)

Imprimé par Groupe d'imprimerie St-Joseph inc.
(Membre de la Corporation St-Joseph)
1165, rue Kenaston, Ottawa (Ontario), Canada K1G 6S1

NORTH ATLANTIC TREATY ORGANISATION

RESEARCH AND TECHNOLOGY ORGANISATION
BP 25 • 7 RUE ANCELLE
F-92201 NEUILLY-SUR-SEINE CEDEX • FRANCE
Telefax 0(1)55.61.22.99 • E-mail mailbox@rta.nato.int

DISTRIBUTION OF UNCLASSIFIED
RTO PUBLICATIONS

NATO's Research and Technology Organisation (RTO) holds limited quantities of some of its recent publications and those of the former AGARD (Advisory Group for Aerospace Research & Development of NATO), and these may be available for purchase in hard copy form. For more information, write or send a telefax to the address given above. **Please do not telephone**.

Further copies are sometimes available from the National Distribution Centres listed below. If you wish to receive all RTO publications, or just those relating to one or more specific RTO Panels, they may be willing to include you (or your organisation) in their distribution.

RTO and AGARD publications may be purchased from the Sales Agencies listed below, in photocopy or microfiche form. Original copies of some publications may be available from CASI.

NATIONAL DISTRIBUTION CENTRES

BELGIUM
Etat-Major de la Défense
Département d'Etat-Major Stratégie
ACOS-STRAT-STE – Coord. RTO
Quartier Reine Elisabeth
Rue d'Evère, B-1140 Bruxelles

CANADA
DRDKIM2
Knowledge Resources Librarian
Defence R&D Canada
Department of National Defence
305 Rideau Street, 9th Floor
Ottawa, Ontario K1A 0K2

CZECH REPUBLIC
DIC Czech Republic-NATO RTO
VTÚL a PVO Praha
Mladoboleslavská ul.
Praha 9, 197 06, Česká republika

DENMARK
Danish Defence Research Establishment
Ryvangs Allé 1, P.O. Box 2715
DK-2100 Copenhagen Ø

FRANCE
O.N.E.R.A. (ISP)
29 Avenue de la Division Leclerc
BP 72, 92322 Châtillon Cedex

GERMANY
Streitkräfteamt / Abteilung III
Fachinformationszentrum der Bundeswehr, (FIZBw)
Friedrich-Ebert-Allee 34
D-53113 Bonn

GREECE (Point of Contact)
Defence Industry & Research General Directorate
Research Directorate
Fakinos Base Camp
S.T.G. 1020
Holargos, Athens

HUNGARY
Department for Scientific Analysis
Institute of Military Technology
Ministry of Defence
H-1525 Budapest P O Box 26

ICELAND
Director of Aviation
c/o Flugrad
Reykjavik

ITALY
Centro di Documentazione Tecnico-Scientifica della Difesa
Via XX Settembre 123a
00187 Roma

LUXEMBOURG
See Belgium

NETHERLANDS
Royal Netherlands Military Academy Library
P.O. Box 90.002
4800 PA Breda

NORWAY
Norwegian Defence Research Establishment
Attn: Biblioteket
P.O. Box 25, NO-2007 Kjeller

POLAND
Armament Policy Department
218 Niepodleglosci Av.
00-911 Warsaw

PORTUGAL
Estado Maior da Força Aérea
SDFA - Centro de Documentação
Alfragide
P-2720 Amadora

SPAIN
INTA (RTO/AGARD Publications)
Carretera de Torrejón a Ajalvir, Pk.4
28850 Torrejón de Ardoz - Madrid

TURKEY
Millî Savunma Başkanliği (MSB)
ARGE Dairesi Başkanliği (MSB)
06650 Bakanliklar - Ankara

UNITED KINGDOM
Dstl Knowledge Services
Kentigern House, Room 2246
65 Brown Street
Glasgow G2 8EX

UNITED STATES
NASA Center for AeroSpace Information (CASI)
Parkway Center
7121 Standard Drive
Hanover, MD 21076-1320

SALES AGENCIES

NASA Center for AeroSpace Information (CASI)
Parkway Center
7121 Standard Drive
Hanover, MD 21076-1320
United States

The British Library Document Supply Centre
Boston Spa, Wetherby
West Yorkshire LS23 7BQ
United Kingdom

Canada Institute for Scientific and Technical Information (CISTI)
National Research Council
Acquisitions
Montreal Road, Building M-55
Ottawa K1A 0S2, Canada

Requests for RTO or AGARD documents should include the word 'RTO' or 'AGARD', as appropriate, followed by the serial number (for example AGARD-AG-315). Collateral information such as title and publication date is desirable. Full bibliographical references and abstracts of RTO and AGARD publications are given in the following journals:

Scientific and Technical Aerospace Reports (STAR)
STAR is available on-line at the following uniform resource locator:
http://www.sti.nasa.gov/Pubs/star/Star.html
STAR is published by CASI for the NASA Scientific and Technical Information (STI) Program
STI Program Office, MS 157A
NASA Langley Research Center
Hampton, Virginia 23681-0001
United States

Government Reports Announcements & Index (GRA&I)
published by the National Technical Information Service
Springfield
Virginia 22161
United States
(also available online in the NTIS Bibliographic Database or on CD-ROM)

Printed by St. Joseph Print Group Inc.
(A St. Joseph Corporation Company)
1165 Kenaston Street, Ottawa, Ontario, Canada K1G 6S1

ISBN 92-837-0032-5