



NRL/MR/5540--14-9541

A Framework for Event Prioritization in Cyber Network Defense

ANYA KIM
MYONG H. KANG
JIM Z. LUO
ALEX VELAZQUEZ

*Center for High Assurance Computer Systems
Information Technology Division*

July 15, 2014

Approved for public release; distribution is unlimited.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing this collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden to Department of Defense, Washington Headquarters Services, Directorate for Information Operations and Reports (0704-0188), 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to any penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number. **PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ADDRESS.**

1. REPORT DATE (DD-MM-YYYY) 15-07-2014			2. REPORT TYPE Memorandum Report		3. DATES COVERED (From - To)	
4. TITLE AND SUBTITLE A Framework for Event Prioritization in Cyber Network Defense					5a. CONTRACT NUMBER	
					5b. GRANT NUMBER	
					5c. PROGRAM ELEMENT NUMBER	
6. AUTHOR(S) Anya Kim, Myong H. Kang, Jim Z. Luo, and Alex Velazquez					5d. PROJECT NUMBER	
					5e. TASK NUMBER	
					5f. WORK UNIT NUMBER 55-6804-14	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Research Laboratory, Code 5542 4555 Overlook Avenue, SW Washington, DC 20375-5320					8. PERFORMING ORGANIZATION REPORT NUMBER NRL/MR/5540--14-9541	
9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES) Office of Naval Research One Liberty Center 875 N. Randolph St., Suite 1425 Arlington, VA 22203-1995					10. SPONSOR / MONITOR'S ACRONYM(S) ONR	
					11. SPONSOR / MONITOR'S REPORT NUMBER(S)	
12. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release; distribution is unlimited.						
13. SUPPLEMENTARY NOTES						
14. ABSTRACT Cyber warriors need to make quick, effective decisions regarding cyber events: namely, which events should be addressed first (i.e., event triage/prioritization) and what should be done with them (i.e., event response). Events should be triaged based on the potential damage they have to important assets and the overall mission. This enables cyber warriors to better protect critical missions by focusing on high priority events. Existing tools used in current practice do not provide such effective event prioritization. Effective event prioritization should include factors such as the importance of the host, the vulnerabilities of the host, network connectivity, as well as details of the event itself. We developed a framework to prioritize events based on the potential damage that each event can incur to important hosts and missions. The framework gathers, fuses, and integrates relevant information from other security tools and databases for automated event prioritization. We implemented our framework as a flexible, extensible, customizable, and user-friendly tool called ACCEPT.						
15. SUBJECT TERMS Cyber security Cyber network defense Event priority						
16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON	
a. REPORT	b. ABSTRACT	c. THIS PAGE			Anya Kim	
Unclassified Unlimited	Unclassified Unlimited	Unclassified Unlimited	Unclassified Unlimited	46	19b. TELEPHONE NUMBER (include area code) (202) 767-6698	

Table of Contents

Abstract.....	1
1. Introduction.....	2
2. Event Priority Framework.....	3
2.1. Host Exposure.....	7
2.2. Asset Criticality	8
2.3. Host Importance.....	10
2.4. Connectivity.....	11
2.5. Comprehensive Host Importance.....	12
2.6. Event Impact.....	13
2.7. Event Effect on Host	13
2.8. Comprehensive Event Effect	14
2.9. Event Clustering.....	14
2.10. Event Propagation Potential.....	15
2.11. Event Correlation	15
2.12. Event Priority Value.....	16
2.13. Related Work	16
3. ACCEPT – Architecture, Operation and Features.....	17
3.1. ACCEPT Data Sources	17
3.2. Tool Implementation of EP Framework Modules.....	18
3.3. Tool Features	18
3.4. Graphical vs. Tabular View	19
3.5. Concepts and Parameters.....	22
3.6. Data Management Features	23
3.7. The Concepts and Parameters feature Strengths of the Tool	26
3.8. Implementation Details	27
3.9. Tools Testing and Validation.....	27
4. Conclusion and Future Work	28
5. Acknowledgements.....	29
6. References.....	30
Appendix A. Asset Criticality.....	32
Appendix B. Modified TOPSIS Algorithm	35
Original TOPSIS Method (Technique for Order Preference by Similarity to Ideal Solution)	35
Modified TOPSIS Method.....	37
Appendix C. Host Exposure Module.....	39
Overview.....	39
Inputs to the Module.....	39
Calculations.....	40
Final Score	43
Weighting.....	43

A Framework for Event Prioritization in Cyber Network Defense

Abstract

Cyber warriors are responsible for defending their organization's cyber assets against an overwhelming number of cyber events. They need to make quick, effective decisions regarding events: namely, which events should be addressed first (i.e., event triage/prioritization) and what should be done with them (i.e., event response). Events should be triaged based on the potential damage they have to important assets and the overall mission. This enables cyber warriors to better protect critical missions by focusing on high priority events. Existing tools used in current practice do not provide such effective event prioritization. Currently, event prioritization is performed by looking at the event severity value provided by security appliances. This value does not always consider host or network information and as such, provides no information about the damage it can have to the target host or the organization's missions. For event prioritization to reflect effects of the events on hosts and missions, it should include factors such as the importance of the host, the vulnerabilities of the host, the network connectivity, as well as details of the event itself. Most of this information is already available in various tools and published data, but they are hard to utilize because currently they must be manually processed.

We developed a framework to prioritize events based on the potential damage that each event can incur to important hosts and missions. The framework gathers, fuses, and integrates available and relevant information from other security tools and vulnerability databases for automated event prioritization. High impact events that target or affect important hosts will have a higher priority than other events. This enables cyber warriors to focus on these events, resulting in critical missions being protected.

We implemented our framework as a flexible, extensible, customizable, and user-friendly tool called ACCEPT. ACCEPT can reduce the workload of cyber warriors and serve as a force multiplier by allowing them to focus on addressing the most serious events.

1. Introduction

Cyber network defense is a continuous battle. Security breaches continue to threaten our networks and cyber infrastructure. High-tech attackers, sometimes backed by well-funded nation states wage increasingly sophisticated cyber attacks [1] [2]. Cyber warriors are responsible for ensuring critical information is safely guarded and essential mission applications are running in the face of these cyber attacks and security breaches. However, lack of manpower and an overwhelming number of attacks make cyber network defense an extremely difficult task [3]. For example, within the Department of Navy (DoN), a relatively small number of personnel protect almost one million DoN cyber assets from hundreds of thousands of events on a daily bases [4]. While many tools are employed, these tools lack the information to enable efficient addressing of these events. These issues result in the cyber warrior playing a constant game of catch-up. The average time to resolve an attack for most organizations requires 14 days, according to the Ponemon Institute [5].

Cyber warriors need better methods and actionable information to help them efficiently address cyber events to protect important assets and the missions they support. In particular, a way to triage incoming events is in urgent need so cyber warriors know which events should be neutralized first to protect important assets and missions. Event prioritization should consider additional factors such as importance of the host, the vulnerabilities of the host, the network connectivity, as well as details of the event itself. Event prioritization should be consistent and standardized within an organization.

Current practice in event triage uses the event severity values provided by or inferred from network monitoring tools to gauge event priority. However, this event severity is a value that is assigned to each event independent of information about the host, and thus cannot reflect the actual impact the event on the specific host that is being targeted. Furthermore, it is difficult to compare event severity values from different sources because different security tools have different scales and meaning associated with their severity values. These limitations prohibit current event triage attempts from being effective.

Cyber warriors may apply some domain expertise to event data to further triage events, but this requires a high level of situational awareness and manual information fusion that are difficult to keep up with the current tempo of operations. The cyber warrior has many tools available that provide the necessary information; vulnerability scanners provide information about what vulnerabilities exist on each host and the host information such as its list of applications, its function, and the organization it belongs to can be found through various systems. The lack of automated tools requires manual gathering, compilation, and fusion of information. Oftentimes, cyber warriors must employ out-of-band techniques such as making a phone call to get additional information. Also, the manual process creates inconsistencies among cyber warriors, allowing important events to slip through the cracks.

What cyber warriors need is an integrated event prioritization framework that determines true event priority based on the events' effects to the hosts. It should be able to process data from different databases and tools to obtain the required factors. It should properly scale and fuse the different data into usable knowledge for accurate event prioritization. At the same time, cyber warriors should be able to understand the event prioritization decisions without being overwhelmed with too much information. This would allow the warrior to spend their limited time and resources implementing responses to the events that deserve their immediate attention.

In this paper, we present our event prioritization framework for cyber defense. We also describe our tool, A Configurable Cyber Event Prioritization Tool (ACCEPT) that implements the framework.

2. Event Priority Framework

Our event priority framework is composed of quantified numeric values associated with events that can be used to determine which events need to be addressed first. It is a complex concept that needs to take a large number of factors into account to produce accurate information. Our approach is to create a hierarchy of contributing concepts that encapsulate those factors and calculate the final event priority value. Intermediate concepts are defined in a way that will be meaningful to the cyber warriors to provide insights into the rationale behind the calculations. To understand the rationale, cyber warriors should be able to drill down into the details of the calculation and understand where the values come from and what they are based on. Conceptualizing the intermediate steps allows the cyber warrior to easily follow the calculation from the raw data all the way to the final event priority value. This is important because event prioritization can be highly subjective. Cyber warriors may prioritize events differently depending on their organization, role, situation, personal preference, etc. If cyber warriors disagree with the event prioritization results, they should have enough understanding to be able to drill down, determine where the disagreement lies, and then reconfigure it. The framework must be flexible and customizable enough to conform to the requirements of different organizations and operational contexts.

Raw data fed into the framework are normalized into numeric values for the appropriate contributing concepts. The intermediate concepts are encapsulated into factors. They are combined according to mathematical formulas to form higher-level factors and then into the final event prioritization value. The formulas are parameterized and the weights of the different factors can be tweaked and customized to suite the organizational needs. They can also be viewed independently to provide insights into the events, the hosts, and the network. Figure 1 provides a conceptual diagram of our event priority framework. We now provide a brief overview of the contributing concepts depicted in the diagram.

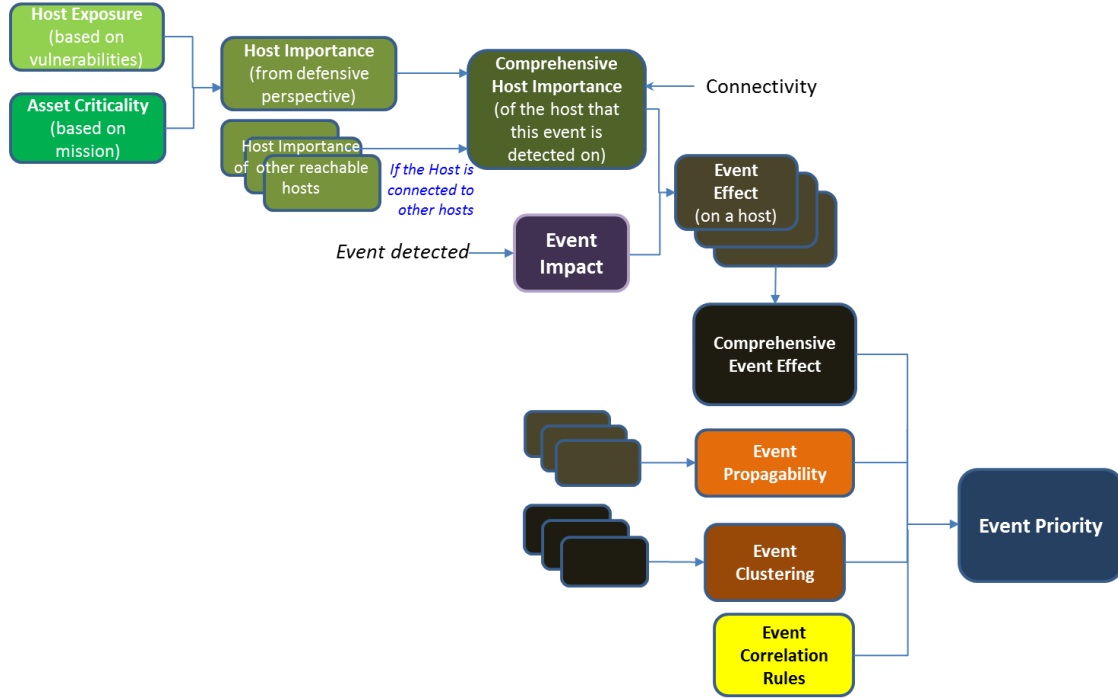


Figure 1. Overview of the Event Prioritization Framework

Before we can assess the potential damage that can be resulted from an event, we need to determine how important each host is (Figure 2a). We cannot expect that the same event will result in the same kind of damage for all kinds of hosts. The host that supports critical missions has more to lose from the attack. The host that has more vulnerabilities (or vulnerabilities with more severe consequences) would be potentially more susceptible to the attack and suffer more consequences. Therefore, both the asset criticality of the host and its exposure to vulnerabilities are important factors in determining the effect of the impact of the event. We call these characteristics collectively the *Host Importance*. We break these concepts down into confidentiality, integrity, and availability (CIA) dimensions to provide more detailed information and better prioritization to the cyber warrior. This also allows us to take advantage of the National Vulnerability Database (NVD) [6] that also describes vulnerabilities in terms of these three dimensions.

Hosts, in general, do not operate in isolation. They are deployed on networks where hosts in close proximity influence each other. We define *Connectivity* as an abstract concept that quantifies the degree of influence that an attack on one host can have on other hosts. The *Comprehensive Host Importance* concept takes into account connectivity and the isolated host importance of nearby hosts (Figure 2b).

Once the *Event Impact* on a host is determined, it is combined with the *Comprehensive Host Importance* to calculate the *Event Effect* (Figure 2c). For example, an event that has a high confidentiality impact will affect a host with a high confidentiality requirement differently than a host with a low confidentiality requirement. Because *Event Impact* is measured independent of the host, it does not change. However, the effect of the event (*Event Effect*) that targets the high confidentiality host will be greater.

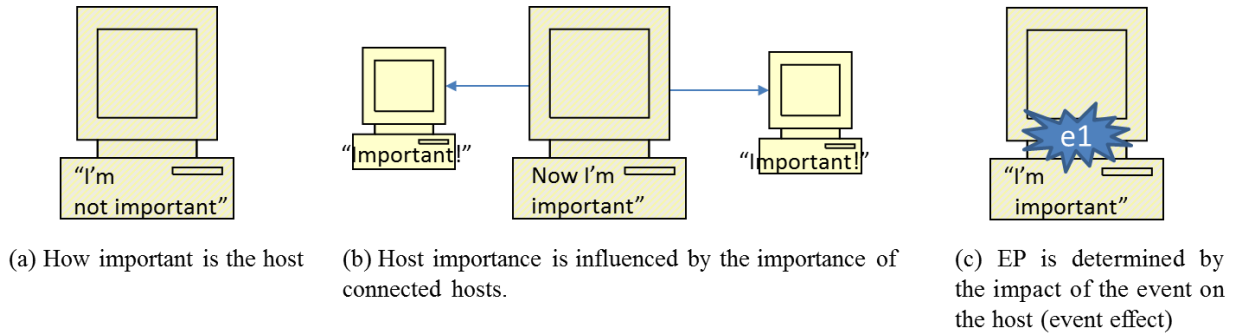


Figure 2. Concepts of Host Importance, Connected Hosts, and Event Impact

In some cases, the same event can target multiple hosts simultaneously. Then, the *Event Effect* of all the targeted hosts for that event needs to be combined to obtain the cumulative *Comprehensive Event Effect* (Figure 3). When an event target just one host its *Comprehensive Event Effect* is the same as its *Event Effect*. *Comprehensive Event Effect* is the most important concept in determining event priority in our framework.

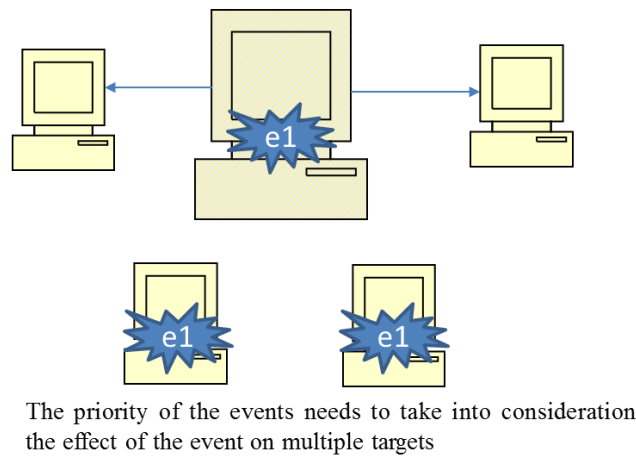
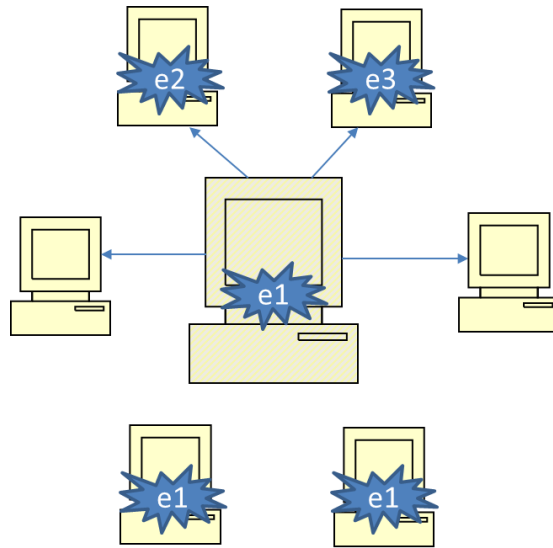


Figure 3. Event e1 targeting multiple hosts

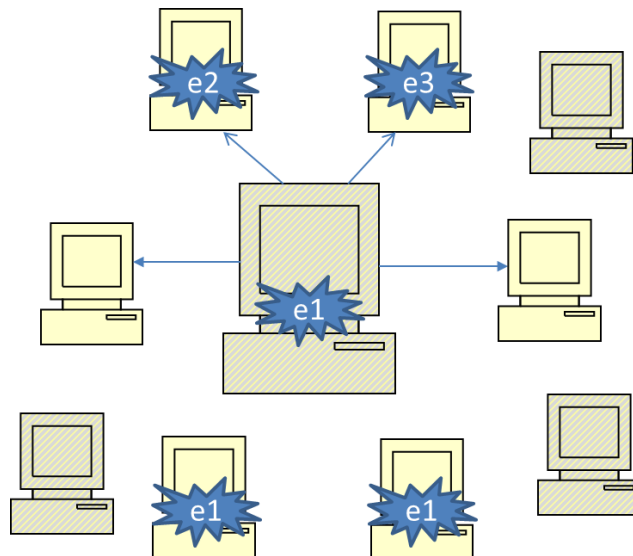
At this point, the confidentiality, integrity, and availability aspects are combined into a single preliminary event priority value that can be compared against the event priority values of other events. However, this is not always sufficient and other factors must also be considered. Connectivity plays a role not only for calculating *Comprehensive Host Importance*, but also for the effects of events. Large number of events in a localized part of the network may indicate a concerted attack and may require more attention. We call this concept *Event Clustering* where event priority is raised slightly by different events occurring on the same host, or other events on connected hosts (Figure 4).



Events are influenced by other nearby events

Figure 4. Event e1 with clusters of events nearby

Another minor factor is *Event Propagability*. In general, events spread to other hosts through connections. For example, the Print Spooler Service Impersonation vulnerability (CVE-2010-2729) spreads via networks when systems share a printer over the network [7]. However, certain events like worms or viruses do propagate regardless of connectivity (e.g., via email). These events have the potential to affect a huge number of hosts that share the same configuration (exploited by the event) as the target host [8]. Figure 5 shows hosts sharing the same vulnerable configuration as the target host, with the same shading. Therefore, the priority of these kinds of events should be raised regardless of connectivity. The potential effect of these events is determined by the number of hosts that share the same configuration, the impact of the event, as well as the host importance value of these hosts.



Event types that affect other hosts regardless of connectivity must take into consideration the potential targets

Figure 5. Events that propagate and affect other hosts

In addition to processing raw data input from the security appliances, the framework should also be able to take into account the domain knowledge of the cyber warriors. The *Event Correlation* concept allows cyber warriors to increase or decrease the Event Priority by creating customized rules. Once written, the rules are automatically processed by a reasoning engine without human intervention.

The diagram in Figure 1 summarizes the Event Priority Framework. The next subsections provide a more detailed explanation of each factor in the event prioritization framework.

2.1. Host Exposure

Host Exposure assists users in determining the likelihood of a host being the target of an attack, of an attack being successful, and the potential damage of attacks using CIA as specific measurements. The Host Exposure factor calculates a numerical measure of risk for potential attacks on the host. It provides quantitative values for vulnerabilities on the host that can be realized by an event. Host Exposure (HE) is calculated for the three dimensions of confidentiality (HE_C), integrity (HE_I), and availability (HE_A). The combined host exposure value is calculated as the Euclidean norm of the components values:

$$HE(h_i) := (HE_C(h_i), HE_I(h_i), HE_A(h_i)) \text{ for host } h_i$$

The input to this module is the list of vulnerabilities for each host represented in Common Vulnerabilities and Exposures (CVE) format using the Common Vulnerability Scoring System (CVSS) [9]. It is available in the National Vulnerability Database (NVD) [6]. The vulnerabilities and their CVE values can be obtained from running a vulnerability scanner such as Nessus [10]. We parsed the CVE base scores obtained from the NVD database to determine the impact of vulnerabilities on the host in terms of confidentiality, integrity, and availability. The base scores by themselves did not provide an accurate measure of a host's potential exposure for one main reason. The base score formulas were designed to compare two different vulnerabilities and not two different hosts with multiple (aggregated) vulnerabilities. As a result, many of the underlying elements used to determine base scores become obfuscated by the CVSS equations. Without breaking a CVE score into its components, there is no straightforward way to determine if a combination of specific vulnerabilities will result in a complete compromise to the host system. In order to best evaluate a host, our framework takes into account how a set of elements of each CVE score can be combined in order to completely compromise the host instead of simply using the base score.

We separate the CVE score for each vulnerability v into multiple sub-scores: Impact to Confidentiality, Impact to Integrity, Impact to Availability, Access Vectors, Complexity, and Authentication Requirements. These values, along with the attack surface (i.e., how many vulnerabilities are on the system/host) are combined to determining the host's potential exposure to attacks. This measure takes into account the number of vulnerabilities (attack surface, $ATT_{C,I,A}$) and their damage potential (e.g., high confidentiality loss, medium integrity impact) either collectively (max impact, $IC_{C,I,A}$) or individually (max vulnerability, $MV_{C,I,A}$). For example, a host with 10 vulnerabilities can have a maximum impact of the combination of the effects of all 10 vulnerabilities, or only an individual vulnerability may be realized. When either the number of vulnerabilities is small or the individual vulnerability impact is small, the host

is not highly exposed to vulnerabilities. However, a large attack surface or vulnerabilities with high impact will result in the host exposure value being high.

The Host Exposure values are calculated as follows:

$$HE_{C,I,A}(h_i) = IC_{C,I,A}(h_i) * (10 - y) * MI_{C,I,A}(h_i)^{W_{MI}} * MV(h_i)^{W_{MV}} * ACA(h_i)^{W_{ACA}} + ATT_{C,I,A}(h_i, x, y)$$

Where $IC_{C,I,A}$ is the Initial Compromise Level, $MI_{C,I,A}$ is the Max Impact, MV is the Max Vulnerability, ACA is the combined Access Vector/Complexity/Authentication, and $ATT_{C,I,A}$ is the Attack Surface with two parameters x and y . W_{MI} is the weight of the max input, W_{MV} is the weight of the max vulnerability, W_{ACA} is the weight of the combined Access Vector/Complexity/Authentication.

To simplify the formulas, we use a vector notation throughout the paper. For example, $HE_{C,I,A}(h_i)$ implies that the Host Importance value is computed for each of the three dimensions of confidentiality, integrity, and availability separately for host h_i .

Host exposure values are scaled from 0 to 10. Since each component used to modify the final score ranges from 0 to 1, weighting that specific component without affecting the rest of the function is accomplished using multiple powers. Weights are assigned from 0 to 1, where 0 represents no effect on the final score ($w^0 = 1$) and 1 represents full effect ($w^1 = w$). This value is the relative rate of each component on the final score. So, a weight of 0.5 is half as effective as a weight of 1.

A detailed explanation of the Host Exposure algorithm and each term in the equation is provided in Appendix C.

2.2. Asset Criticality

Asset criticality (AC) represents the degree of support the asset provides to the function of the organization in terms of confidentiality, integrity, and availability. It is a way of acknowledging that not all cyber assets are equally important. From a military perspective, it is defined in terms of how crucial the asset is to the success of the mission. Even in the commercial sector, there are various missions or objectives of the organization that of which some assets play a more crucial part than others. Unfortunately, an asset's contribution level to a mission is not directly observable, nor is there any tools readily providing this information [11]. Thus, asset criticality needs to be estimated from measurable attributes of the assets. There has been little work done in this area, but the work that has been done has been either looked at assets from the attacker's (instead of the defender's) perspective, require manual user input in determining the value of each asset, take into account too few characteristics, or simply examine dependencies and network topology between asset [12] [13] [14] [15] [16] [17]. Others such as [18] evaluate assets based on characteristics of the computing environment, division of classes of systems, and policy goals of the system in terms of its confidentiality, integrity, and availability. This information is ultimately used to determine the cost of selecting an intrusion response. While the characteristics used to evaluate assets are similar to ours, their approach is a manual process that entails expert knowledge and a high degree of judgment. We need an AC measurement that incorporates all the relevant attributes, and can be done with minimal user involvement.

Different categories of assets support missions in different ways. For example, host machines, routers, and printers all provide different types of mission support. These different categories require different attributes to be measured to determine asset criticality. We focus on calculating the AC values for host machines specifically.

We calculate the numerical AC values first by calculating a base AC value, then adjusting the value for data dependencies, and finally, adjusting the AC values for user-defined AC attributes. The base AC value is obtained by identifying attributes that contribute to estimating the criticality of the asset. These attributes are: 1) criticality level of the applications on the host, 2) network sensitivity level, 3) value of data on the asset, 4) importance of the command that the host belongs to, and 5) the host operating system. These attributes were selected because we felt that they are relevant to the criticality of a host, and the data was available from the existing tools. When considering which things to use as attributes, we needed to consider how well they could be used to infer the criticality of an asset as well as if the values were available from any of the security tools being used. For example, “what mission apps are running on the host” may be a good candidate in terms of its indication of asset criticality, but that information is not available from any of the tools that were used by our cyber warriors. On the other hand, the type of applications on an asset provides a measure of the host’s purpose. For example, if it has an Apache server on it, we can assume that it is a web server. The sensitivity level of the data on the machine is also important in determining the criticality of an asset; all other things being equal, a machine on a classified network should have more priority than a machine on a public network. The value of data can also be used to help distinguish assets. Mainly, this helps raise the value of hosts that may be on public networks, have no ‘important’ software on them, but contain a lot of private data such as personally identifiable information (PII). Machines in an HR department may fit this description. This information is not directly available, but can be inferred by correlating employee databases, organizational charts, and machine ownership information available from system administrators. For military purposes, the command that a host belongs to also helps determine the machine’s purpose as well as its level of mission support. Finally, our cyber warriors stated that most critical assets run a specific kind of operating system, so we added that to our set of attributes. One thing to stress is that, by itself, not one attribute can accurately determine machine criticality. However, taken together, the resulting value gives a good representation of the criticality of an asset.

Some attributes, such as the importance of the command that the host belongs to, need to be converted into a numerical value. We can get the command that a host belongs to as raw data. We then need to calculate the numerical score. For these types of attributes, we provide a table that translates the corresponding attributes value into numerical values for each of the confidentiality, integrity, and availability dimensions. For example, we assign a confidentiality value of 5 to 5th Fleet, and a 3 to NRL. The use of a table allows cyber warriors to easily change these values, so that for instance, NRL can have a numeric value of 2, by changing the value in the table. We scale the values of attributes and results so that the asset criticality component value for confidentiality, integrity, and availability ranges from 0 to 10.

Once the confidentiality, integrity, and availability values for the individual attributes are determined, we apply a multi-attribute decision making algorithm [19] [20] [21], called Technique for Order of Preference by Similarity to Ideal Solution (TOPSIS), to obtain a numerical value that represents the base AC score for each component. This algorithm assumes the existence of a theoretically high asset (whose attributes all have maximum values) and a theoretically low asset (whose attributes have all minimum

values). The criticality of each asset is obtained by measuring its relative closeness to the theoretically high asset (using Euclidean distance). We modified TOPSIS so that it would work more efficiently in a setting where hosts and their associated attributes can be added, removed or changed frequently.

$$AC(h_i) = f(apps, network, data, command, os)$$

where f is the modified TOPSIS function

This is done for the confidentiality component (AC_C), the integrity component (AC_I), and the availability (AC_A) component. The combined asset criticality value is calculated as the Euclidean norm of the three dimensions:

$$AC(h_i) := (AC_C(h_i), AC_I(h_i), AC_A(h_i)) \text{ for host } h_i$$

In the next step, we examine dependency relationships among hosts. An asset that is seemingly not critical based on the values of the attributes may provide input to other critical assets. Since these critical assets depend on the host for input, this provider host should also be considered critical. This step attempts to capture this concept of criticality based on dependencies. The base AC value for each provider asset is adjusted by considering the strength of the dependencies between provider host and dependent host, the number of dependent assets, the AC value of the dependent assets, as well as the initial AC value of the provider asset. The reason that dependency is not factored into the base score is so that critical hosts without dependencies are not penalized in the initial AC calculations, but rather hosts with dependencies AC values are adjusted later. The formula to adjust the AC values for provider hosts is:

$$newAC_{C,I,A}(h_i) = \min\{10, AC_{C,I,A}(h_i) + \log_{10}\left[\sum_k AC_{C,I,A}(h_k) \cdot w_k\right]\}$$

where $AC(h_k)$ is the AC value of dependent host h_k and w_k is the strength of its dependency

Finally, these interim scores may be overridden by user-defined values. Despite what the results of the AC calculation may show, sometimes cyber warriors will have information about a host that contradicts the results of the calculation. In that case, we want to allow the cyber warrior to override the calculated AC value. This step allows for these user-defined values. The AC values are also scaled from 0 to 10.

A complete/detailed explanation of the asset criticality algorithm is provided in Appendix A, the modified TOPSIS approach we developed and implemented is presented in Appendix B.

2.3. Host Importance

Before events even occur, we can calculate how important a host is from the cyber defense perspective by considering its host exposure and asset criticality values. Host Importance (HI) is also broken down into the three dimensions of confidentiality (HI_C), integrity (HI_I), and availability (HI_A). The combined host importance value is calculated as the Euclidean norm of the components values:

$$HI(h_i) := (HI_C(h_i), HI_I(h_i), HI_A(h_i)) \text{ for host } h_i$$

A highly mission critical host with no vulnerabilities theoretically may not need be defended since it cannot be attacked, but the fact that it is mission critical still makes it an important target from both an

attack and defense perspective. A host with millions of vulnerabilities may seem to require a lot of defense but if it is not a critical asset, it is probably not an important host to defend. Therefore, we need to combine these two factors in such a way that the Host Importance can be accurately determined. The Host Importance algorithm combines asset criticality and host exposure as follows:

$$HI_{C,I,A}(h_i) = AC_{C,I,A}(h_i) \cdot acBase^{\alpha/\beta}$$

Where $\alpha = exposureFactor \cdot HE_{C,I,A}(h_i)$ and $\beta = dampeningFactor \cdot AC_{C,I,A}(h_i)$

In the formula, *acBase*, *exposureFactor*, and *dampeningFactor* are parameterized weights. The weight *acBase* determines the curve of the graph, and should be greater than 1, otherwise HI and AC will be identical. The weight α determines how much Host Exposure affects Host Importance regardless of the value of Asset Criticality. Weight β determines the degree of how much Host Exposure affects Host Importance as Asset Criticality values increase. Higher β values result in Host Exposure having more effect on Host Importance as Asset Criticality values rise. In general, the ratio of (*exposureFactor/dampeningFactor*) determines how much Host Exposure affects the Host Importance relative to Asset Criticality. While both should affect Host Importance, we want Asset Criticality to affect it more; a highly critical asset that has no vulnerability exposure is still a very important asset and so its Host Importance value should be close to the AC value. However, a highly exposed, non-critical asset should have a relatively low importance. We determined that this was represented best when Host Importance and Asset Criticality had a nearly linear relationship. When Host Exposure is held constant, Host Importance and Asset Criticality have a nearly linear relationship since the AC factor in the exponent is almost negligible compared to the AC multiplication component. For default values, we set *acBase* as 1.1, *exposureFactor* as 0.8, and *dampeningFactor* as 0.9. Host Importance values are scaled from 0 to 10.

2.4. Connectivity

Hosts on the network do not operate in isolation and therefore cannot be considered by themselves. For example, the Asset Criticality factor captures the dependency relationship between hosts because this dependency can make a host important. However, hosts that do not directly interact can still have an influence on each other. We define connectivity as an abstract concept that quantifies the degree of influence a host can have on another host. It provides information on how easily an event can spread through the network from the target host and affect other hosts. This information assists in determining event priority from a global perspective. We define connectivity in the abstract and leave it up to the user to determine how to implement it. For example, users can determine connectivity at the network level: a host is always fully connected to itself, strongly connected to hosts within the same subnet, less strongly connected between subnets, etc. VLANs and firewall rules can also be added to further distinguish connectivity. Connectivity can also be defined using attack graphs and attack paths. The decision of how to implement connectivity can be based partially on what tools and system information are available to the cyber warrior. Hosts are considered strongly connected to hosts that are immediately upstream of them on possible attack paths. Regardless of the implementation, the concept of connectivity is important in order to include the influence of the network topology and connected hosts in determining host importance, event importance, and event priority overall.

We represent connectivity as an $n \times n$ matrix M , where n represent hosts in the cyber domain and $m_{i,j}$ is the *connectivity value* that represents the strength of the connection from host h_i to host h_j . By default this value ranges from 0 to 1, where 0 means no connection and 1 denotes fully connected. As per our definition of connectivity, we leave the implementation details of connectivity to the user. They can define connectivity in terms of the network level, attack paths, network hops, etc.

The connectivity value is directional. The value of $m_{i,j}$ is not necessarily equal to the value of $m_{j,i}$. The connectivity value is not transitive. The value in the cell $m_{i,j}$ and $m_{j,k}$ is not related to the value in the cell $m_{i,k}$. The connectivity value is not broken down into confidentiality, integrity, and availability components. This is done to simplify the calculations.

2.5. Comprehensive Host Importance

In section 2.3, we defined Host Importance in isolation from other hosts. Now, using connectivity, we can define Comprehensive Host Importance. Comprehensive Host Importance (CHI) takes into account the isolated Host Importance of the host itself as well as the isolated Host Importance values of other connected hosts using the connectivity values from the matrix.

Again, the values are computed across the dimensions of confidentiality (CHI_C), integrity (CHI_I), and availability (CHI_A). The combined comprehensive host importance value is calculated as the Euclidean norm of the components values:

$$CHI(h_i) := (CHI_C(h_i), CHI_I(h_i), CHI_A(h_i)) \text{ for host } h_i$$

Comprehensive Host Importance is calculated according to the following formula:

$$CHI_{C,I,A}(h_i) = HI_{C,I,A}(h_i) + connWeight \cdot \sum_j (HI_{C,I,A}(g_j) \cdot m_{i,j})$$

where *connWeight* is a weight and $m_{i,j}$ is the connectivity value in the matrix between host h_i and hosts g_j .

We assign a connectivity weight (*connWeight*) to the formula to determine the level of influence connected hosts have on the Comprehensive Host Importance. The default connectivity weight is set low at 0.1 but can be changed by the cyber warrior. The Host Importance values of the connected hosts are multiplied by their connection values in the matrix to ascertain the influence of the connected hosts. For example, a connected host with high Host Importance value may make the original host more important. However, if they are weakly connected, then the effect will be decreased.

In theory a network can contain millions of hosts that are connected. This potentially creates an extremely large theoretical maximum for the Comprehensive Host Importance. The value for highly connected networks can become artificially inflated and convergent in a way that does not make sense. The connectivity component of Comprehensive Host Importance can overwhelm the contribution of the isolated host importance so that individual hosts become indistinguishable. To address this issue we provide a limit to the number of connections that are added to the calculation. This limit is set by default to the highest 10 connections. Limiting the number of connections to consider also improves performance.

While connectivity and comprehensive host importance can be calculated before events occur and not in real time, performance is still useful to consider. Using these default values, the range of values that the Comprehensive Host Importance can have is 0 to 20. The connectivity component of the score can contribute at most 10 points to the overall value.

2.6. Event Impact

Event Impact quantifies the amount of damage an event can have. It looks at the underlying vulnerability that the event exploits and compares it to the list of vulnerabilities on the host. As mentioned earlier, the CVE entries published by the National Vulnerability Database (NVD) provide impact scores for vulnerabilities in terms of impact to confidentiality, impact to integrity, and impact to availability [6]. These values give insight into the amount of damage an event could cause if the vulnerability were to be exploited. However, this value is determined regardless of the target host. So while it provides more information than the simple event severity generated by security appliances, it is of limited usefulness by itself. For example, an event identified as an *Adobe Reader Integer Overflow DoS* attack when cross-referenced with the underlying vulnerability (CVE ID: CVE-2009-1856) in the NVD database provides details as to its level of impact [22]. This information provides us with a more accurate estimate of the potential impact an event can have on a host than the event severity value. In particular, CVE-2009-1856 results in a complete loss of confidentiality, integrity, and availability. It is important to note that even though we have an Event Impact score, we cannot know what the actual effect to the host is without knowing the CIA requirements of the target host.

The impact values from the CVS are qualitative ranging from *low* to *complete*. We convert these values into numeric values, ranging from 0 to 10, so they can be processed in the framework. The reason we separate out the impact value into impact to confidentiality, integrity, and availability is so that more detailed and accurate assessment of an impact may be made. For example, a vulnerability with overall high impact may seem to have a high effect on a host. However, if we can break down the impact so that we know that it affects the confidentiality and integrity of a host, but not its availability, and we know that the target host is a public web server that has no confidential or critical information, or any other confidential or integrity requirements, then we know that the actual effect of the vulnerability will be minimal to that specific host. Given this, an event e_l that exploits CVE-2009-1856 will have an Event Impact (EI) as follows:

$$EI(e_l) = (EI_C(e_l) = 10, EI_I(e_l) = 10, EI_A(e_l) = 10)$$

In the case where no vulnerability is provided with the event, we use the default event severity values as provided by the security monitoring tool. In this case, the default event severity value is used for event impact for all three components of CIA. This will result in a much lower fidelity, however, we expect that most of the more important events will have vulnerability information attached.

2.7. Event Effect on Host

Once the Event Impact is determined, we look at the effect of the event on the targeted host. Event Effect combines Event Impact values with the Comprehensive Host Importance values. This tells us that an event on a specific host is important because it has a specified impact (e.g., complete loss of confidentiality, partial loss of integrity) to a host with certain characteristics (e.g., high confidentiality requirement, many confidentiality vulnerabilities). Event Effect (EE) is calculated for the confidentiality

component, the integrity component, and the availability component. The combined Event Effect value is calculated as the Euclidean norm of the components values:

$$EE(h_i, e_l) := (EE_C(h_i, e_l), EE_I(h_i, e_l), EE_A(h_i, e_l)) \text{ of event } e_l \text{ detected on host } h_i$$

Event Effect is calculated as follows:

$$EE_{C,I,A}(h_i, e_l) = eventImpactWeight \cdot EI(e_l) + hostImportanceWeight \cdot CHI_{C,I,A}(h_i)$$

The formula is a simple additive function since both the impact of the event and the comprehensive host importance value of the host positively affect the Event Effect. Weights *eventImpactWeight* and *hostImportanceWeight* can determine the importance of the Event Impact and the Comprehensive Host Importance, respectively. We determined that the (Comprehensive) Host Importance was more important and set our default weights accordingly as *eventImpactWeight* = 0.5 and *hostImportanceWeight* = 1.0. Using these values, the Event Effect can have values ranging from 0 to 15.

2.8. Comprehensive Event Effect

If the event targets more than one host, we calculate the effect of the event on all the target hosts by summing up the individual Event Effect values on each target host to obtain the Comprehensive Event Effect (CEE):

$$CEE(e_l) = \sum_i EE(h_i, e_l) \text{ where } e_l \text{ is an event that targets (multiple) hosts } h_i.$$

If an event targets only one host, then its Comprehensive Event Importance (CEE) value is identical to its Event Effect (EE) value.

At this point, we no longer maintain separate calculations for the dimensions of Confidentiality, Integrity, and Availability. These separate dimensions were maintained to justify the calculated values. However, in the end, we want to provide the cyber warrior with one number per event that they can use to compare the priority of many events simultaneously.

Similar to the Comprehensive Host Importance, it is difficult to determine a theoretical maximum for the Comprehensive Event Effect since it is possible that an event can target a large number of hosts on the network. We want to ensure that events with low Event Effects targeting a large number of hosts do not create a large comprehensive Event Effect that could dwarf events with high Event Effects that target a single host. For these reasons, we again limit the number of target hosts that are added to the calculation. Our default limit is the top 10 target hosts. Using this limit, our value ranges for the Comprehensive Event Importance range from 0 to 30.

Comprehensive Event Effect is the major factor that drives Event Priority. We now introduce a few more factors that can affect the priority of an event.

2.9. Event Clustering

Cyber events are not random distributions. The attacks that are of concern are the deliberate ones crafted to achieve a specific goal. Events are clustered around a few connected hosts are likely to be related to a concerted effort and should be prioritized higher. Even if clustered events are indeed unrelated, multiple

events can cause more damage as a whole than the sum of their parts. These events could cause more damage than if they were occurring in isolation and should also be prioritized higher.

We call this factor Event Clustering. Event Clustering (EC) is obtained by computing the Comprehensive Event Effect of *other* events on hosts reachable from the target host. Hosts are connected to themselves, so other events on the same host are also included in the event clustering calculation. For an event e_l , its Event Clustering (EC) value (looking at other surrounding events) is computed as:

$$EC(e_l) = \sum_{\{r\},\{m\}} CEE(h_r, e_m) \cdot m_{i,r}$$

where h_r are hosts that can be reached from the target host h_i , e_m are other events on those reachable hosts, and $CEE(h_r, e_m)$ is the Comprehensive Event Effect of these events on the reachable hosts, while $m_{i,r}$ is the connection value between target host h_i and reachable host h_r .

Similar to the Comprehensive Event Importance, we put a limit on the number of clustered events to be examined. The default value is 10, providing a range of values from 0 to 30 for Event Clustering.

2.10. Event Propagation Potential

An additional characteristic that we consider is the potential propagation of an event, independent of connectivity. In other words, how many other hosts that have not been infected by an event can be susceptible to it because they share the same configuration and vulnerabilities as the hosts with the event on it? For example, a worm exploit specifically targeting Windows 2007 machines propagated by email can very quickly spread to other hosts regardless of connectivity. Event Propagability determines the potential damage to other hosts on the network with the same configuration as the target host for these types of events. It is calculated by summing the Comprehensive Event Effect that these same-configuration hosts would have if the event were to propagate to them:

$$ER(e_l) = \sum_v CEE(h_v)$$

Again, we put a limit on the number of propagable hosts to be examined. The default value is 10, providing a range of values from 0 to 30 for Event Propagability.

2.11. Event Correlation

Sometimes, the importance of two or more seemingly unrelated events may be more than the sum of their individual importance. Events can compound each other's effectiveness. The importance of events may need to be raised given the presences of other events or conditions. Groups of event may have low importance individually, but they represent a coordinated attack that should be addressed with high priority. Cyber warriors will have domain knowledge about specific attack patterns that security appliances are not aware of. For example, an attack may be launched on a web server on the DMZ, and then a separate event detected on a DB server behind a firewall that provides information to the web server. The events occur on separate networks, and are detected by two different IDS as different events. However, cyber warriors with the appropriate domain knowledge will know that this is an exfiltration

attack to get information out of the DB server using the Web server. These two events need to be correlated and the event effect should be raised appropriately.

The Event Correlation module allows domain knowledge of cyber warriors to be codified, automatically evaluated, and taken into account for event periodization. This is done by using the Drools Fusion [23] rule-based engine to detect and process these types of patterns. The engine uses a series of “when...” components to recognize patterns and detect the situation, and a series of “then...” statements to take an appropriate action. The result is that new EP values are assigned to one or more of the correlated events as defined in the action. Example actions may include cases when two events are determined to be related, one or both events’ priority values are raised, or when several events are correlated into one new event, and the new event is assigned a priority value. The Event Correlation module can lessen the workload of a cyber warrior by applying domain knowledge but in an automatic and efficient manner.

2.12. Event Priority Value

After all the modules have been calculated, the final event priority is computed in the following manner. If there is no event correlation, the Event Priority is calculated by combining the Comprehensive Event Effect (CEE) of the event with its Event Propagability (ER) and Event Clustering (EC) modifiers:

$$EP(e_l) = CEE(e_l) + clusteringWeight \cdot EC(e_l) + propagabilityWeight \cdot ER(e_l)$$

where *clusteringWeight* and *propagabilityWeight* are parameters used to weight the Clustering and Propagability values so they do not overwhelm the effect of the event on the target host. On the other hand, if the event is a correlated event obtained from the event correlation rules, the actions in the Drools rule-base determines what the event’s EP value would be.

2.13. Related Work

There has been some work in similar areas of determining appropriate course of actions for specific events or increase situational awareness by examining the effects of attacks on missions. Even though they do not prioritize events, the ideas and concepts discussed in these papers are comparable to the methods to determine event priority.

The work of Sawilla *et al.* [24] uses dependency attack graphs to describe complex network attacks. Dependency attack graphs are more concise and efficient than other attack graphs because they do not attempt to encode all possible states of the system. However, even dependency attack graphs that map local networks can grow unwieldy very quickly when having to model the many events on a network. On the other hand, this approach may be a potential method to model connectivity in our framework.

Lewis *et al.* [25] developed a reference model for impact assessment and situational assessment, where the factors include missions, services, IT assets, network connections, known vulnerabilities, safeguards, cyber alerts, attack categories and partial models of attacks. They use Certainty Factors to assess the situation. This is a very high-level reference model that tells you what missions are impacted at what stage of an attack, but provide no quantitative measurements. Other works on assessing situational awareness have also been identified [15] [26] [27]. All of these models use a narrow set of factors to assess the security situation and do not provide any actual or quantitative methods for prioritizing events.

The work that is most similar to ours is the Cyber Threat Prioritization Implementation Framework from SEI [28]. They point out that analysts have diverse approaches in event prioritization and use a narrow scope of factors. Their framework breaks down cyber threats into three core components: the likelihood of threat actors executing attacks, the impact threats have on an organization's business, and the risk threats pose because of an organization's known vulnerabilities. While this is a useful approach, the framework is described at a very high level and does not provide specific or quantitative measurements that can be applied against the hundreds of thousands of events that occur on the network. There are no methods presented as to how to combine these factors and how to get the data.

The implantation of our framework, introduced in the next section, was implemented in a real operational setting with data from actual security tools to calculate event priority for the numerous events that are detected on the system in real time.

3. ACCEPT – Architecture, Operation and Features

ACCEPT is the implementation of our EP framework. It captures, analyzes and processes data obtained from various security tools and encapsulates the specific event prioritization steps into independent modules. Each module in the tool corresponds to a factor in the EP framework described in the previous section. ACCEPT is an extensible framework in that each module is parameterized and configurable so easy customization. Users can tailor the tool to their organization's specific operational requirements and data availability by reconfiguring the modules or adding their own modules to the framework.

During our research and throughout the tool development process, we collaborated with security people responsible for network monitoring and event handling to derive tool requirements – what data should be displayed, what format should it be displayed in, and what they would do with this information. This tool is a result of the recognition of a need for event prioritization, and collaboration between our research and the requirements of the cyber warriors who defend our networks.

3.1. ACCEPT Data Sources

ACCEPT leverages support from the cyber environment that is being monitored by using its tools (e.g., sensors, firewalls, vulnerability scanners) to collect and provide the raw data and act as data sources to ACCEPT. The specific cyber environment we used is from the Navy Cyber Defense Research Laboratory, a research laboratory supporting the Navy Computer Network Defense Service Provider. The data sources used in the tool are [29] [30] [31] [13] [32] [10] [6]:

- McAfee HBSS/ePO rollup data for each host
- Asset Publishing Service (APS) module from HBSS/ePO suite for host data
- Asset Configuration Compliance Module (ACCM) data for inventory of host configurations
- Vulnerability scanning results from ACAS/NESSUS
- Vulnerability scanning results from SCCVI/RETINA
- Event data from a Security Information and Event Management (SIEM) tool such as NetIQ Sentinel (aggregated from sources such as Snort, HBSS/ePO, McAfee NSM)
- Common Vulnerability Scoring System (CVSS) scores from the NVD database.

Since much of the information we gather is from different sources, the source formats were varied (e.g., XML format, CSV format) and were not usable as is. Our tool brings all this data together into a cohesive and processible manner. For example, the CVE data provides scores that obfuscate the impact of the individual factors. We had to separate out the factors and recombined them in a way that made sense of event priority. In addition, the host data was extracted and recombined in a way to assess asset criticality.

If ACCEPT is used in a different environment, the tools and available data sources may differ. In these cases, some additional preprocessing of the data may be required before the raw data can be used as input to ACCEPT.

3.2. Tool Implementation of EP Framework Modules

Each factor of the EP framework (Figure 5) is implemented as a module in the ACCEPT tool. Since the NVD database describes impact in terms of three factors: confidentiality, integrity, and availability, we also describe most of the modules in terms of these three factors as well as a combined factor that takes the vector magnitude of these three values. For example, asset criticality is described in terms of the host's confidentiality, integrity and availability requirements for mission support. The fourth and final AC value is the vector magnitude of these three values. This allows us to say that, for example, a weapons system and a database system may both be mission critical (based on the vector magnitude), but that the weapons system has a higher requirement for integrity, whereas the database system may have a high requirement for confidentiality. The individual confidentiality, integrity, and availability values are preserved and used in module calculations up to the Comprehensive Event Effects module. After that, the combined value is used as the output of each remaining module so that the final Event Priority value does not provide separate scores for confidentiality, integrity, and availability.

3.3. Tool Features

ACCEPT has two modes of operation – real and simulation. The modes have separate databases for their respective hosts, vulnerabilities, and events. The real mode accepts host, vulnerability and event data directly from other security tools. This data is parsed and combined, then used as input into the various modules. Each module is calculated using this data until the event priority values are obtained.

In the simulation mode, we allow users to model the configuration and behavior of the hosts. Users also have the option of using the entire NVD database, a subset, or creating their own vulnerability tables. The tool simulates events – security behavior – based on the specified host configuration and vulnerabilities and generates event priority values for these events. The simulation mode has multiple purposes: 1) it provides a higher level of abstraction for low level cyber events, thus providing a meaningful understanding in the cyber domain, 2) it provides an interface identical to the real mode allowing users to get familiar with the system before going operational, 3) it can be used to test how accurately changes to the modules or their parameters reflect the EP calculation, and 4) it can facilitate users' understanding of the factors that contribute to event priority and enable them to gain faith in the inner workings and results of the tool. In the simulation mode, users can understand how changes in configuration or connectivity affect the various factors. Or they can inject theoretical events and observe how they affect the network.

Both modes support the graphical view, tabular view, concepts and parameterization, and data input features that are described below and are represented as tabs in our tool.

3.4. Graphical vs. Tabular View

We provide users with both a graphical view and tabular view of the data. Both views not only show event priority information, but also capture the justification behind how the value was obtained.

The graphical view provides both a host-centric view and an event-centric view of the current situation in real time. The host-centric view provides a list of the most important hosts and determines event priority from these hosts' point of view (Host-based event priority). Clicking on any of these hosts depicts the host and its connectivity relationship to other hosts, as well as the highest priority events (from the host-based perspective). Rolling over a host provides a popup screen that offers justification as to a host's importance, showing its asset criticality values and host exposure values as well as additional information.

The graphical view in Figure 6 shows the graphical view of hosts annotated with their internal IP addresses. The circles in various colors represent events with different event priority values. The event-centric view provides a list of the highest priority events and displays them in graphical form as shown in Figure 7.

The tabular view provides information about the *hosts*, *software*, *vulnerabilities*, and *events*. The list of *hosts* shows all the hosts within the organization and their comprehensive host importance values (Figure 8). The values of the main factor that comprise comprehensive host importance - asset criticality, host exposure, and host importance - are also shown. For each host, links are provided for the list of software detected on the host, the results of the vulnerability scan on the host, the list of hosts connected to the host, and any events detected on the host. The list of *software* and list of *vulnerabilities* provide a list of all the software and vulnerabilities respectively detected on the entire network, as well as what hosts they are associated with. The list of *events* shows the list of events in the network, their event priority values, as well as values of factors that make up event priority, such as event effect, propagability, and clustering (Figure 9). Entries also have a link to the underlying vulnerability exploited by the event, hosts targeted by the event, list of hosts connected to target hosts, etc.

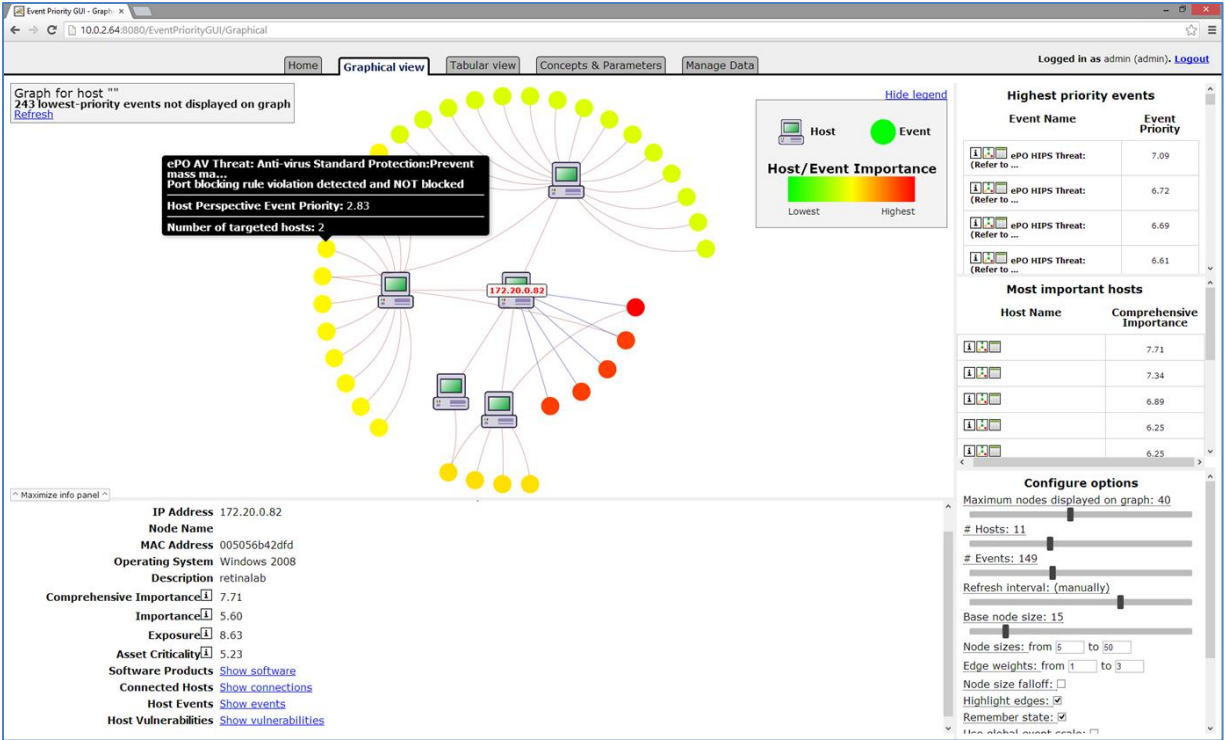


Figure 6. Graphical view of the ACCEPT tool (Host Perspective)

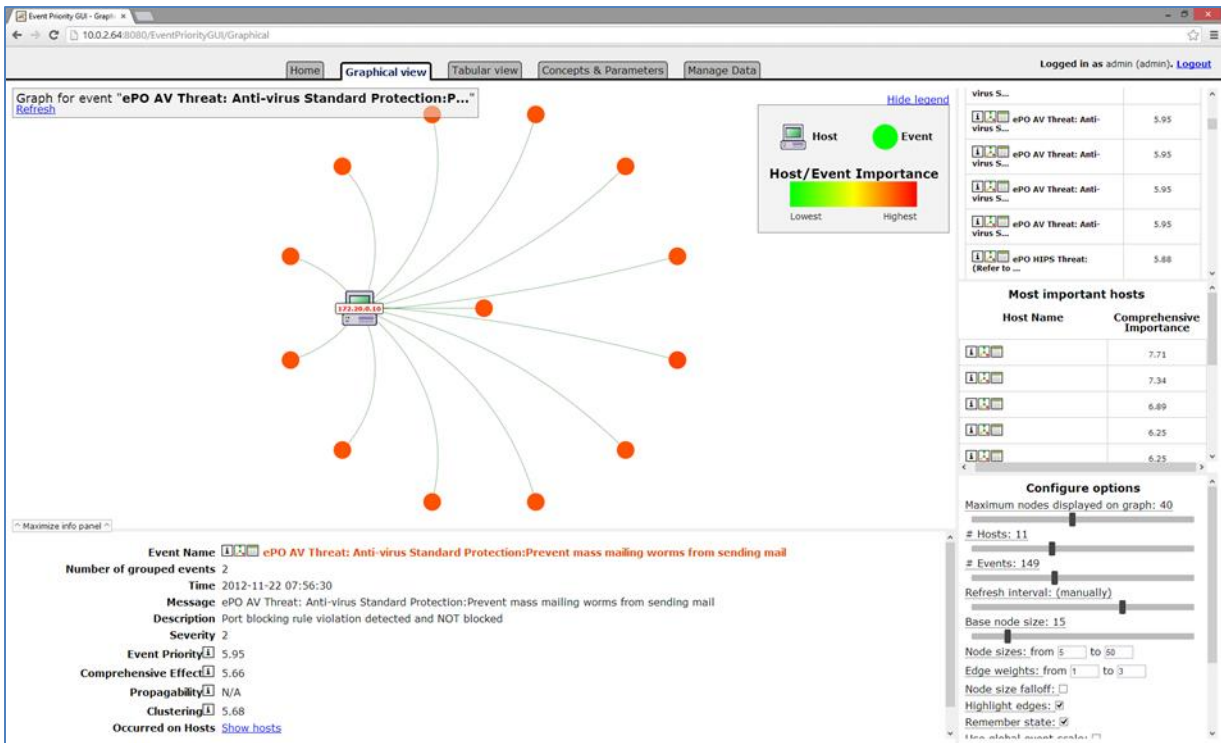


Figure 7. Graphical view of the ACCEPT tool (Network Perspective)

Host IP	Host Perspective	Comprehensive Host Importance ¹				Host Exposure ¹				Asset Criticality ¹				Host Description	Host Software	Host Vulnerabilities	Host Connectivity				
		Combined	C	I	A	Combined	C	I	A	Combined	C	I	A								
172.20.0.82	View_Graph	7.71	6.66	8.83	7.47	5.6	4.61	6.4	5.64	8.63	8.63	8.63	8.63	5.23	4.1	6.13	5.26	retinalab	Software (5)	Vulnerabilities (2)	Connected Hosts (14)
172.20.0.10	View_Graph	7.34	6.28	8.48	7.1	5.25	4.25	6.06	5.29	4.72	5.8	5.8	5.8	5.19	4.06	6.09	5.22	NULL	Software (1)	Vulnerabilities (0)	Connected Hosts (14)
172.20.0.55	View_Graph	6.89	6.14	7.89	6.51	4.81	4.11	5.5	4.73	0.0	0.0	0.0	0.0	5.36	4.58	6.13	5.26	N/A	Software (8)	Vulnerabilities (0)	Connected Hosts (14)
172.20.0.12	View_Graph	6.25	5.26	7.89	5.22	4.21	3.28	5.5	3.49	0.0	0.0	0.0	0.0	4.69	3.65	6.13	3.89	N/A	Software (2)	Vulnerabilities (0)	Connected Hosts (14)
172.20.0.11	View_Graph	6.25	5.26	7.89	5.22	4.21	3.28	5.5	3.49	0.0	0.0	0.0	0.0	4.69	3.65	6.13	3.89	N/A	Software (4)	Vulnerabilities (0)	Connected Hosts (14)
172.20.0.247	View_Graph	6.17	4.98	7.89	5.22	4.14	3.01	5.5	3.49	0.0	0.0	0.0	0.0	4.61	3.35	6.13	3.89	N/A	Software (8)	Vulnerabilities (0)	Connected Hosts (14)
172.20.1.149	View_Graph	5.95	4.44	6.63	6.51	4.98	3.71	5.53	5.49	2.06	0.26	0.26	7.36	5.23	4.1	6.13	5.26	N/A	Software (4)	Vulnerabilities (0)	Connected Hosts (14)
172.20.1.51	View_Graph	5.65	4.42	6.6	5.71	4.7	3.68	5.5	4.73	0.0	0.0	0.0	0.0	5.23	4.1	6.13	5.26	N/A	Software (44)	Vulnerabilities (0)	Connected Hosts (14)
172.20.1.33	View_Graph	5.65	4.42	6.6	5.71	4.7	3.68	5.5	4.73	0.0	0.0	0.0	0.0	5.23	4.1	6.13	5.26	N/A	Software (47)	Vulnerabilities (0)	Connected Hosts (14)
172.20.2.5	View_Graph	5.54	4.42	6.46	5.56	4.7	3.68	5.5	4.73	0.0	0.0	0.0	0.0	5.23	4.1	6.13	5.26	N/A	Software (0)	Vulnerabilities (0)	Connected Hosts (14)
172.20.2.6	View_Graph	5.54	4.42	6.46	5.56	4.7	3.68	5.5	4.73	0.0	0.0	0.0	0.0	5.23	4.1	6.13	5.26	N/A	Software (0)	Vulnerabilities (0)	Connected Hosts (14)
172.20.0.9	View_Graph	5.46	6.53	5.78	3.64	3.48	4.49	3.49	1.99	0.0	0.0	0.0	0.0	3.87	5.0	3.89	2.21	N/A	Software (1)	Vulnerabilities (0)	Connected Hosts (14)
172.20.0.240	View_Graph	4.05	5.28	3.66	2.79	2.19	3.3	1.47	1.18	2.82	2.88	4.8	2.88	2.08	3.35	0.95	0.95	(no description provided)	Software (8)	Vulnerabilities (2)	Connected Hosts (14)
172.20.0.54	View_Graph	4.02	4.98	4.2	2.45	2.14	3.01	1.99	0.86	0.0	0.0	0.0	0.0	2.38	3.35	2.21	0.95	N/A	Software (0)	Vulnerabilities (1)	Connected Hosts (14)
172.20.2.7	View_Graph	3.72	4.4	3.49	3.16	2.98	3.66	2.68	2.45	6.03	6.13	6.13	9.33	2.38	3.35	2.21	0.95	N/A	Software (0)	Vulnerabilities (14)	Connected Hosts (14)
172.20.0.230	View_Graph	3.65	4.98	3.01	2.45	1.87	3.01	0.86	0.86	0.0	0.0	0.0	0.0	2.08	3.35	0.95	0.95	N/A	Software (0)	Vulnerabilities (0)	Connected Hosts (14)
172.20.0.235	View_Graph	3.65	4.98	3.01	2.45	1.87	3.01	0.86	0.86	0.0	0.0	0.0	0.0	2.08	3.35	0.95	0.95	N/A	Software (0)	Vulnerabilities (0)	Connected Hosts (14)
172.20.0.216	View_Graph	3.65	4.98	3.01	2.45	1.87	3.01	0.86	0.86	0.0	0.0	0.0	0.0	2.08	3.35	0.95	0.95	N/A	Software (0)	Vulnerabilities (0)	Connected Hosts (14)
172.20.0.226	View_Graph	3.65	4.98	3.01	2.45	1.87	3.01	0.86	0.86	0.0	0.0	0.0	0.0	2.08	3.35	0.95	0.95	B12R227	Software (0)	Vulnerabilities (0)	Connected Hosts (14)
172.20.0.226	View_Graph	3.65	4.98	3.01	2.45	1.87	3.01	0.86	0.86	0.0	0.0	0.0	0.0	2.08	3.35	0.95	0.95	Rob Hobson KNET System	Software (8)	Vulnerabilities (0)	Connected Hosts (14)
172.20.0.205	View_Graph	3.65	4.98	3.01	2.45	1.87	3.01	0.86	0.86	0.0	0.0	0.0	0.0	2.08	3.35	0.95	0.95	N/A	Software (7)	Vulnerabilities (0)	Connected Hosts (14)

Figure 8. Tabular view of hosts in ACCEPT

Event ID	Network Perspective	Grouped Events	Event Priority	Time	Message	Description	Severity	Propagability	Clustering	Comprehensive Event Effect ¹				Event Vulnerabilities	Event Hosts	Event Connectivity
										Combined	C	I	A			
D3544606-18EA-1030-881B-0050568A536A	View_Graph	2	7.09	2012-11-22 09:17:37	ePO HIPS Threat: (Refer to KB article...	TCP Port Scan	3.0	N/A	4.91	6.84	6.33	7.42	6.73	Vulnerabilities (0)	Event Hosts (2)	Connected Events (283)
D3544606-18EA-1030-8A7A-0050568A536A	View_Graph	2	6.72	2012-11-22 09:44:14	ePO HIPS Threat: (Refer to KB article...	TCP Port Scan	3.0	N/A	5.61	6.43	6.07	6.95	6.26	Vulnerabilities (0)	Event Hosts (2)	Connected Events (283)
4AFC4418-104F-1030-888A-0050568A0008	View_Graph	2	6.69	2012-11-22 09:16:51	ePO HIPS Threat: (Refer to KB article...	TCP Port Scan	3.0	N/A	5.17	6.43	6.07	6.95	6.26	Vulnerabilities (0)	Event Hosts (2)	Connected Events (283)
D3544606-18EA-1030-878E-0050568A536A	View_Graph	4	6.61	2012-11-22 09:15:25	ePO HIPS Threat: (Refer to KB article...	TCP Port Scan	3.0	N/A	3.56	6.43	6.07	6.95	6.26	Vulnerabilities (0)	Event Hosts (2)	Connected Events (283)
D3544606-18EA-1030-88CF-0050568A536A	View_Graph	2	6.59	2012-11-22 09:23:38	ePO HIPS Threat: (Refer to KB article...	TCP Port Scan	3.0	N/A	3.18	6.43	6.07	6.95	6.26	Vulnerabilities (0)	Event Hosts (2)	Connected Events (283)
D3544606-18EA-1030-8A9A-0050568A536A	View_Graph	2	6.14	2012-11-22 08:01:54	ePO AV Threat: Anti-virus Standard P...	Port blocking rule violation det...	2.0	N/A	5.83	5.84	5.33	6.42	5.73	Vulnerabilities (0)	Event Hosts (2)	Connected Events (283)
D3544606-18EA-1030-8A01-0050568A536A	View_Graph	4	5.97	2012-11-22 08:01:39	ePO HIPS Threat: (Refer to KB article...	CMD Tool Access by a Network Awa...	2.0	N/A	2.44	5.84	5.33	6.42	5.73	Vulnerabilities (0)	Event Hosts (1)	Connected Events (283)
D3544606-18EA-1030-A30C-0050568A536A	View_Graph	4	5.97	2012-11-22 04:14:45	ePO HIPS Threat: (Refer to KB article...	Event Log Registry Setting Modified	2.0	N/A	2.44	5.84	5.33	6.42	5.73	Vulnerabilities (0)	Event Hosts (1)	Connected Events (283)
4AFC4415-104F-1030-8ECB-0050568A0008	View_Graph	2	5.97	2012-11-22 05:14:17	ePO HIPS Threat: (Refer to KB article...	Event Log Registry Setting Modified	2.0	N/A	2.44	5.84	5.33	6.42	5.73	Vulnerabilities (0)	Event Hosts (1)	Connected Events (283)
D3544606-18EA-1030-8A01-0050568A0008	View_Graph	2	5.95	2012-11-22 05:14:17	ePO AV Threat: (Refer to KB article...	Port blocking rule violation det...	2.0	N/A	5.68	5.66	5.14	6.24	5.55	Vulnerabilities (0)	Event Hosts (2)	Connected Events (283)

Figure 9. Tabular view of events in ACCEPT

3.5. Concepts and Parameters

The Concepts and Parameters feature allows users to take advantage of ACCEPT’s modular feature. Users can look at the formulas used for each module and tweak the parameters and weights associated with each formula. The concepts and algorithm for each module are explained in detailed so that users can understand each module as well as make educated changes to it, if desired.

As explained in section 2.5, as modules are combined to reach the next module, it becomes extremely difficult to determine a maximum value for each module’s output. Because many of the formulas, such as Comprehensive Host Importance require values from connected hosts to be added, the theoretical maximum value for this formula would be much higher than the median range of values, and would be different for every organization, depending on connectivity values. For example, in an n -host network, the number of connections could theoretically be n (i.e., fully connected network), making values such as Comprehensive Host Importance and Event Clustering potentially very isolated from realistic values. We provide a realistic maximum value for these formulas by limiting the number of connections to look at and providing default values for weights. An example of this is shown in Figure 10.

Figure 10 shows the Event Priority module – the formula used to obtain event priority, as well as the current parameter values used, and the resulting maximum value possible with these parameters. It also shows how users can select the parameters to change their values. Any changes users make are immediately reflected in the page and in subsequent calculations.

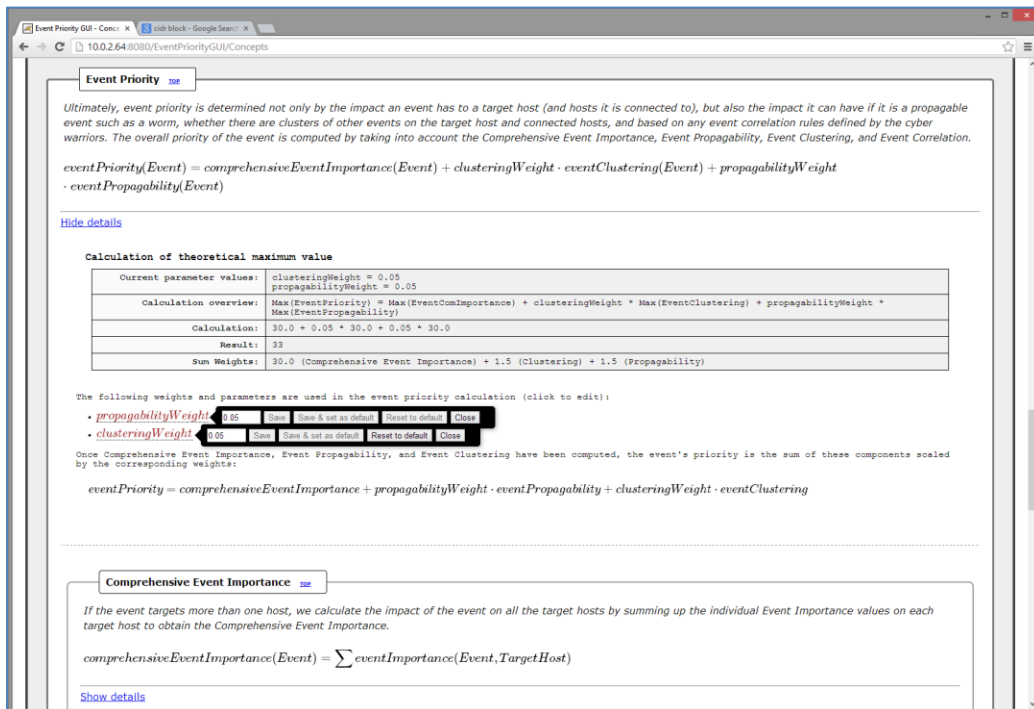


Figure 10. Event priority module customization feature

3.6. Data Management Features

The Manage Data tab provides users a way to enter host, software, vulnerability, and event information. This feature is only available when logged on with administrator privileges. From here users can set the active database to use for main mode using real hosts and events or simulation mode for testing purposes. Existing data entries can be easily removed, and new entries uploaded to the database. Data entries are obtained from the various sources discussed at the beginning of section 3. However, some values need to be converted to a numeric value for calculation. For example, the command (e.g., Fifth fleet, US NAVCENT) that a host belongs to is a factor in determining its asset criticality. We have a table that lists all the commands, and assigns them numerical values (ranging from 1 to 5) according to their strategic/tactical importance. While our tool uses commands as a factor, the flexibility of the framework allows non-military organizations to use it by replacing the commands with divisions (e.g., human resources, payroll, R&D) within its organization, and having a table that translates the divisions into numerical values. The ability to modify or create such tables is provided here.

In addition, users have the option of entering newly found hosts into the main database. For example, initially, host information may be gathered from vulnerability scans and low level host data from other tools. Sometimes there may be conflict among this data due to the way firewalls or other devices mask host IP addresses. Users are given various options for dealing with these unknown IP addresses. Users can also individually edit values of hosts such as the asset criticality values, and host connection values. This is not only useful in real-time mode when cyber warriors may want to artificially raise the value of specific hosts, but also in simulation mode to enable users to better understand the relationship between changes values and event priority. A snapshot of the data management feature is provided in Figure 11.

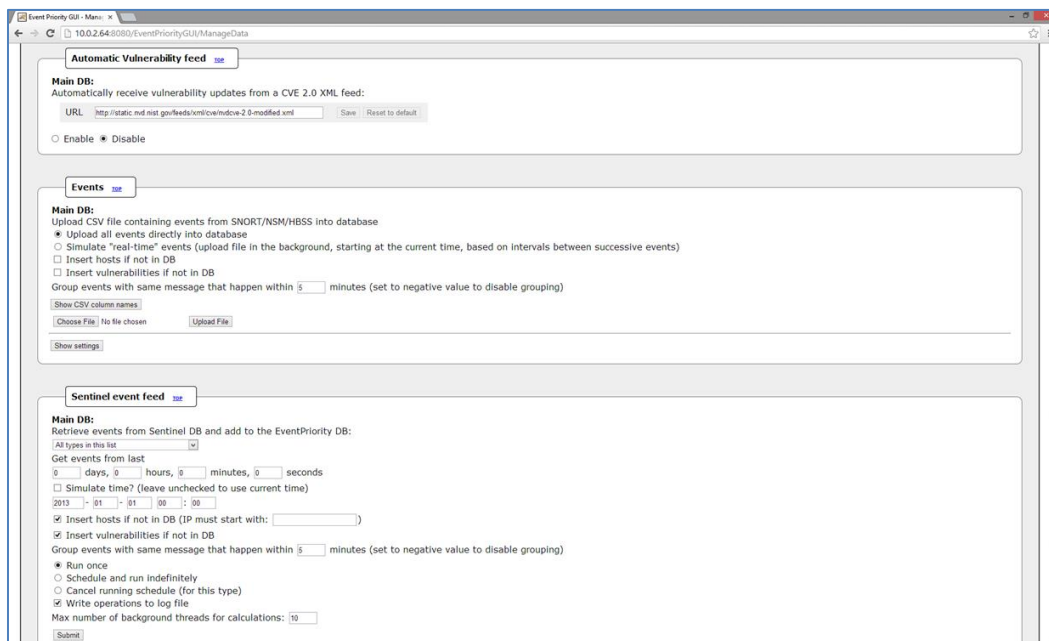


Figure 11. Data management features of ACCEPT

3.6.1. Network Centric and Host Centric View

Up to this point we have described the various factors for event priority and how to combine them from a network point of view. In this section, we describe Host-based Event Priority (HEP). While Network-based Event Prioritization (NEP) prioritizes events based on their effects on the network (considering all hosts), HEP looks to prioritize events from the perspective of a given host. For a specific host of interest (HOI), it examines surrounding events that can cause the most potential damage to that particular host. It provides a different way of using the framework and the tool. Our modular approach enables reorganization of the different /modules to create a different view for the cyber warrior. If the cyber warrior is aware of an important host that needs to be protected at all costs, the HEP viewpoint gives them the ability to see which events can cause the most potential damage to that host. The different viewpoint requires a different way of determining event priority. Given a host, the only events that the host cares about are the ones that can affect it, and only to the degree that they affect that host. Of all the events on the network, the only ones that can potentially affect the HOI are the following types of events:

- Case 1: Events targeting the HOI directly.
- Case 2: Propagable events that target hosts with identical configurations to the HOI and thus could propagate to the HOI regardless of connectivity.
- Case 3: Events detected on hosts that are connected to the HOI. If the events successfully compromise the target host, they can lead to the HOI compromise due to connectivity.
- Case 4: Events detected on hosts that the HOI depends on (for data): the event may not directly impact the HOI but if the provider host is down, the HOI may not be able to provide mission support.

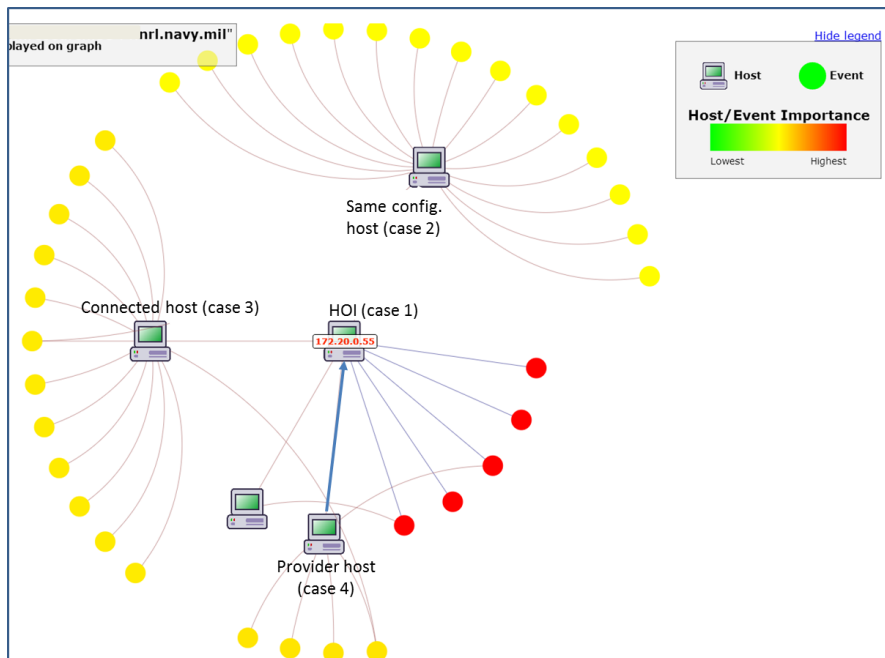


Figure 12. Cases of host perspective events

Of these, only the first case directly affects HOI while the other cases affect HOI indirectly. For example, Figure 12 shows a host perspective view of events. It shows all the events that can affect the HOI (the host depicted in the middle); some events occur directly on the HOI, others occur on different target hosts, but can indirectly affect the HOI. The *event effects* of each of these events are calculated for the target host. Figure 13 shows the framework for calculating Host Perspective Event Priority. It shows Host Perspective Event Priority values calculated for each case, given a HOI. Because it uses a host-centric view, HEP does not consider the importance of other connected hosts or potential damage to other hosts that it is connected to. Thus, combined host importance and combined event effect are not considered in this framework. Host Perspective Event Priority values are calculated for each case as follows:

- Case 1: The Event Effect of each event on the HOI is calculated.
- Case 2: While the event is detected on a machine other than the HOI, this type of event can propagate to the HOI. From the HOI perspective, we want to know the effect of this type of event if it manages to propagate to the HOI. Therefore, we calculate the potential Event Effect of this event on the HOI if it were propagated.
- Case 3: Events occurring on connected hosts, if successful, can lead to compromise of the HOI. We compute the Event Effect of these events on the target host. The stronger the connectivity, the more likely that the event can lead to compromise of the HOI.
- Case 4: Events that occur on provider hosts can disrupt the mission support of the HOI. The degree of event effect on the provider host and the strength of the dependency are proportionate to the importance of the event to the HOI. We calculate the Event Effect of these events on the target host.

After all the event effect values for all the pertinent events have been calculated, the Host Perspective Event Priority value is calculated by using the Event Effect as-is for events that target the host directly, or assigning weights (ω_i) to the indirect Event Effects to create Host Perspective Event Priority values. This allows events of the same type occurring on the HOI, connected hosts, propagable-event hosts, and provider hosts to be prioritized differently. These values are then sorted in order to determine the events that are most important to the HOI.

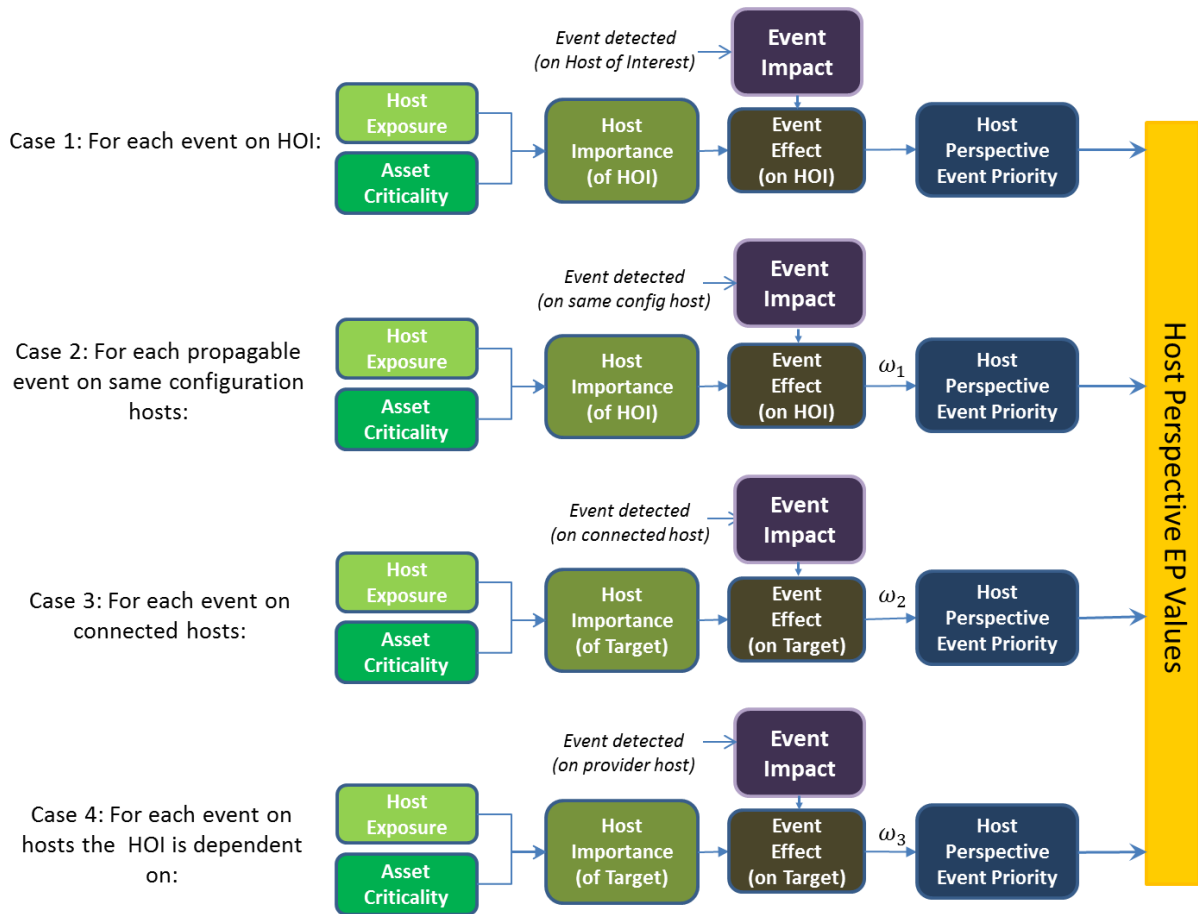


Figure 13. Event prioritization framework for host perspective (ω_i are weights)

3.7. The Concepts and Parameters feature Strengths of the Tool

Besides the simulation capabilities that enable users to answer questions such as ‘how does this event affect the network’ and the customizable features provided by the Concepts and Parameters tab and the Manage Data tab, because ACCEPT combines the previously isolated low level data into actionable information, it can perform tasks beyond event prioritization. Some examples are:

- Can mitigate damage of zero-day attacks

Once a zero-day attack has been detected, ACCEPT can easily find all hosts that have the same underlying configuration required for the attack. Cyber warriors can quickly take steps to strengthen these hosts before the zero-day attack becomes widespread and critical.

- Can further weed out low-level events.

Events generally have vulnerabilities associated with them. Intrusion detection tools provide some correlation and filtering of events. However, currently when events are triggered in the system, the SIEM has no way of knowing if the target host possesses the specific vulnerability. ACCEPT’s cohesive data

integration enables (automatic) checking to see if the host has the given vulnerability, and if not, can rule out the triggered event. Features like these aid in filtering out low-level events and thereby increasing the efficiency of the event prioritization framework. ACCEPT also provides some low level event correlation so that minor and frequent events such as TCP port scans were correlated based on their time. For example, all TCP port scan events that occur within a given time frame on a given host are considered as one event. This not only provides a high level correlation, but also provides less unrelated information cluttering the screen.

- Can be retrofitted to suit different perspectives

Lastly, the modular feature of our tool enables the modules to be rearranged in different ways to obtain other results. For example, a different way that we introduced for using the framework is to view events from a host perspective. While the Network-based EP answers, “which events should I deal with first?” the Host-based EP framework can be used to answer the question, “which events are most detrimental to a given host?” Users can reconfigure the modules further to match the requirements of their own organization.

3.8. Implementation Details

ACCEPT is a Java web application that runs in an Apache Tomcat application server. Its data store is a PostgreSQL database, which it connects to via JDBC. The database contains various tables with information for the Event Priority framework, but also other tables that store information required by the web application (e.g., a "users" table that allows different users to store their customized settings that apply across the application). Dependencies between the tables are represented and enforced via foreign key constraints. The PostgresAdapter class provides methods through which other Java classes can interact with the database. It translates requests into SQL queries, connects to the database, fetches results, and returns them as Java objects.

As values are added or updated (e.g., an event is detected, a host's criticality changes, a parameter is adjusted), computed values need to be recalculated according to the framework's algorithms. For example, as a new event is detected, the event impact on the target host is calculated, the event clustering factor for this event needs to be calculated, and the clustering factor for neighboring events may need to be recalculated to factor in this new event. Hosts, vulnerabilities and events all have their own repository so that atomic changes do not require recalculations of all repositories and can be easily propagated. Recalculation is handled by the Calculator class, which contains implementations of these algorithms. Its methods can be invoked by other classes to trigger a recalculation when changes in the system are detected.

Much of the front-end of the web application is written in HTML/CSS/JavaScript, which is delivered to the client via Java HTTP servlet classes. A servlet is invoked by the client by navigating to a path relative to the application's URL. These servlets call upon the application's back-end classes to assemble an appropriate response to the client's request.

3.9. Tools Testing and Validation

The modular approach of the tool facilitated the testing. Each module was able to be tested and validated separately, before being combined and tested as a whole. The various default weights for each module

Performance testing was also performed and found to be acceptable. For testing, we pulled data in from the Sentinel database every 20 minutes. After low level correlation, this resulted in about 50 new events being entered into the ACCEPT tool, taking a total processing time of 4.5 seconds. During peak times, we would need to insert 500+ events at once into the ACCEPT tool, requiring about 40 seconds of calculations. To enhance performance, we also implemented multi-threaded calculations. This increased the performance even more, so that calculation and insertion of approximately 500 events took about 7 seconds.

With the aid of the Navy subject matter experts, we were able to procure real time data for events and hosts to test our tool. The results of the tool were validated by domain experts. The simulation part of the tool also came in handy for validation purposes, allowing us to create simulated hosts and vulnerabilities, and verify the results of the asset criticality calculations for each host. We were able to generate events and examine the event priority values for correctness as well.

4. Conclusion and Future Work

ACCEPT provides event prioritization that goes beyond looking at just the severity of the event. It is an easily extendable and flexible tool supporting the cyber warrior in the security domain. It relieves the cyber warrior of cumbersome details, and provides them with actionable information regarding events.

To our knowledge, there are no tools out there that can take this data from different sources in different formats, combine it into meaningful information that provides a high-level picture determines accurate event priority. Our solution determines which hosts are important, analyzes events in terms of their impact to these hosts, either directly or by indirect means, and other ways that events can affect critical cyber assets such as important hosts. ACCEPT not only takes input from disparate sources but also creates new intelligence out of the data by combining them in ways that provide situational awareness.

Further design goals include providing better security (authentication based access control), facilitating the addition of event correlation rules, and performing further testing and validation. One limitation regarding this tool is that the accuracy of the results is limited to the information available. The lack of information is two-fold: 1) some information, such as asset criticality is not provided at all by the tools in the environment and has to be estimated by selecting factors that represent the host's mission criticality, and 2) some of the information is dependent on the tools used by each organization. If tools collect different information, or information in different formatting is used, then additional preprocessing may be required. While we devised a scheme to rate the asset criticality of non-host machines such as printers and fax machines, we were not able to implement it in the system because the tools used by the organization did not provide any distinction between host machines and non-host machines.

We expect this tool to be used as part of an organization's security management suite, either as a stand-alone tool or as a component of the SIEM. Data from the SIEM components and other monitoring tools can be pushed to the ACCEPT tool as needed – host-related data that does not change frequently can be pushed less often than events. For example, since standard events in HBSS are reported every 5 minutes, the tool can request new events to be pushed to itself every 5 minutes to coincide with this timeframe.

We see ACCEPT as a way that enables analysts to move from a reactive stance to a proactive posture. It can be used to characterize what is happening on the network and even be used as a predictive tool to drive policy and global cyber strategy. If better low level monitoring and security tools were to be developed that would provide us with even more raw data to examine and correlate, the tool could provide more accurate and sophisticated results.

With the ability to prioritize events based on their effects to critical assets, the next logical step would be to use these values as part of a response system. Our future goal will be to utilize ACCEPT as input to an autonomous response system that can further assist the cyber warrior.

5. Acknowledgements

We would like to thank the members of NRL's Navy Cyber Defense Research Laboratory for their enormous help providing data, explaining concepts, and providing feedback and recommendations with respect to the tool as well as the entire research process.

Last but not least, thank to interns Seth Hitefield and Min Cheol Kim for their invaluable work on the modules and particularly to Mr. Hitefield for writing the content of Appendix C.

6. References

- [1] D. Charles and P. Eckert, "China military hackers persist despite being outed by U.S.," *Reuters (US online Edition)*, pp. <http://www.reuters.com/article/2013/11/06/net-us-usa-china-hacking-idUSBRE9A51AN20131106>, 6 Nov 2013.
- [2] M. Rosenbach, T. Schulz and W. Wagner, "Google Under Attack: The High Cost of Doing Business in China," 19 January 2010. [Online]. Available: <http://www.spiegel.de/international/world/google-under-attack-the-high-cost-of-doing-business-in-china-a-672742.html>. [Accessed 25 Oct 2013].
- [3] Endeca, "Arming the Analyst for Cyber Defense," 2011. [Online]. Available: <http://directory.govloop.com/files/govloop/files/13.pdf>. [Accessed 10 December 2012].
- [4] M. Kagan, "Cyberdefenders Protect Navy Networks," *Military Information Technology (MIT)*, vol. 13, no. 11, December 2009.
- [5] Ponemon Institute, "First Annual Cost of Cybercrime Study, Benchmark Study of U.S. Companies," Sponsored by ArcSight, 2010.
- [6] National Institute of Standards and Technology (NIST), "National Vulnerability Database," [Online]. Available: <http://nvd.nist.gov/>. [Accessed 14 January 2012].
- [7] NIST, "Vulnerability Summary for CVE-2010-2729," [Online]. Available: <http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2010-2729>. [Accessed 2012 2 March].
- [8] Wikipedia, "Stuxnet," [Online]. Available: <http://en.wikipedia.org/wiki/Stuxnet>. [Accessed 16 2013 August].
- [9] P. Mell, K. Scarfone and S. Romanosky, "A Complete Guide to the Common Vulnerability Scoring System Version 2.0," NIST, June 2007.
- [10] Tenable Network Security, "Nessus Vulnerability Scanner," [Online]. Available: <http://www.tenable.com/products/nessus>. [Accessed 29 2010 January].
- [11] A. Kim and M. Kang, "Determining Asset Criticality for Cyber Defense, Memo Report NRL/MR/5540-11-9350," Naval Research Laboratory, Washington DC, 2011.
- [12] Iman Loloei, H. R. Shahriari and A. Sadeghi, "A Model for Asset Valuation in Security Risk," in *20th Iranian Conference on Electrical Engineering, (ICEE2012)*, Tehran, Iran, May 15-17, 2012,.
- [13] I-Assure, "Security Services Appliance," 1 August 2009. [Online]. Available: <http://www.i-assure.com/downloads/Security%20Services%20Appliance.pdf>. [Accessed 20 September 2013].
- [14] Youngja Park, C. Gates and S. C. Gates, "Estimating Asset Sensitivity by Profiling Users," *Computer Security – ESORICS 2013*, vol. 8134, no. Lecture Notes in Computer Science, pp. 94-110, 2013.
- [15] L. Beaudoin and P. Eng, "Asset Valuation Technique for Network Management and Security," in *6th IEEE International Conference on Datamining (ICDMW'06)*, Hong Kong, China, 2006.
- [16] J. R. Goodall, A. D'Amico and J. K. Kopylec, "CAMUS: Automatically Mapping Cyber Assets to Missions and Users," in *Military Communications Conference (MILCOM)*, Boston, 2009.
- [17] R. E. Sawilla and X. Ou, "Identifying Critical Attack Assets in Dependency Attack Graphs," in *Proceedings of the 13th European Symposium on Research in Computer Security*, Malaga, Spain, 2008.
- [18] N. Stakhanova, C. Strasburg, S. Basu and J. S. Wong, "Towards cost-sensitive assessment of intrusion response selection," *Journal of Computer Security*, vol. 20, no. 2-3, pp. 169-198, 2012.
- [19] E. Triantaphyllou, *Multi-criteria Decision Making Methods: A Comparative Study (Applied Optimization)*, Springer, 2010.

- [20] K. P. Yoon and C.-L. Hwang, Multiple Attribute Decision Making: An Introduction (Quantitative Applications in the Social Sciences), SAGE Publications, Inc, 1995.
- [21] G.-H. Tzeng and J.-J. Huang , Multiple Attribute Decision Making: Methods and Applications, Chapman and Hall/CRC; 1 edition , 2011.
- [22] NIST, "Vulnerability Summary for CVE-2009-1856," [Online]. Available: <http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2009-1856>. [Accessed 10 January 2011].
- [23] J Boss Community, "Drools - The Business Logic Integration Platform," [Online]. Available: <http://www.jboss.org/drools/>. [Accessed 10 June 2013].
- [24] R. E. Sawilla and C. Burrell, "Course of Action Recommendations for Practical Network Defense," Defense R&D Canada, Ottawa, 2009.
- [25] L. Lewis, G. Jakobson and J. Buford, "ENABLING CYBER SITUATION AWARENESS, IMPACT ASSESSMENT,," in *Military Communications Conference, MILCOM 2008*, San Diego, CA, 2008.
- [26] M. Heckman, N. Joshi, M. Tylutki, K. Levitt and J. Just, "An Impact Assessment Model for Distributed Adaptive Security Situation Assessment," Defense Advanced Research Projects Agency, Arlington, VA, 2005.
- [27] D. Shen, G. Chen, L. Haynes and E. Blasch, "Strategies Comparison for Game Theoretic Cyber Situational Awareness and Impact Assessment," in *10th International Conference on Information Fusion*, Quebec, Canada, 2007.
- [28] T. Townsend and J. Mcallister, "Implementation Framework - Cyber Threat Prioritization," Software Engineering Institute , Carnegie Mellon University, September 2013.
- [29] "HBSS Host Based Security System," DISA, [Online]. Available: <http://www.disa.mil/Services/Information-Assurance/HBSS>. [Accessed 1 August 2013].
- [30] "HBSS Components," DISA, [Online]. Available: <http://www.disa.mil/Services/Information-Assurance/HBSS/Components>. [Accessed 1 August 2013].
- [31] "ACAS Assured Compliance Assessment Solution," DISA, [Online]. Available: <http://www.disa.mil/Services/Information-Assurance/ACAS>. [Accessed 1 August 2013].
- [32] "Snort," Sourcefire, [Online]. Available: <http://www.snort.org/>. [Accessed 5 Oct 2013].

Appendix A. Asset Criticality

In cyber defense, there are various tools that can tell us about the events, but little is known about the targets. Some tools can provide pieces of information about the hosts. However, there is no standard way to combine this information to get a ‘standardized’ asset criticality (AC) metric that indicates the relative importance of assets. Which assets are more critical and to what degree depends on factors that determine the contribution of an asset to mission success.

The structure of an organization exists to ensure successful mission completion and continuation of operations. Hence, asset criticality should be defined in terms of how crucial the asset is to the success of the mission. Unfortunately, an asset’s contribution level to a mission is not directly observable. Thus, asset criticality needs to be estimated from measurable attributes of the assets that we call *attributes* or *criteria*. We calculate the AC value first by calculating the base AC value using these attributes, then adjusting the score for data dependencies among hosts, and finally, adjusting the AC value for user-defined attributes.

Step 1. Calculate the Base AC Value

The attributes we selected to calculate the base AC value are:

- Criticality level of applications on the host: Through the SIEM tools, we obtain a list of software for each host. The type of s/w available gives us an idea of the machine’s purpose which in turn helps determine its mission criticality. For example, a host with Windows Mail Server on it tells us that the host is probably an email server. We created a table of <Service/application, criticality value {Integrity, Availability}> tuples that associate a machine’s purpose to a numeric criticality value for integrity and availability, ranging from 1 (low) to 5 (High), as shown in Table 1. For example, a File server rates a 5 for integrity and 3 for availability. We do not assign criticality values for confidentiality since the Network sensitivity factor (described below) determines the confidentiality value. These values can be changed for each organization by changing the tables within the ACCEPT Tool.
- Network sensitivity level: Primarily for the use of DoD machines, this value is based on whether the machine is on a public, sensitive, or classified network. We take these network sensitivity values and assign them numeric values of 1, 3, and 5 respectively.
- Value of data on the asset: some hosts have more sensitive or PII information on them than others. The network sensitivity level would address most of these, but some machines not on the sensitive network may still have sensitive data on them, and be considered more critical than others, such as a host machine in the human resources department that contains payroll information. We map the system administrator’s table of <IP address, machine owners (name)> to the organizational chart of <employee name, role>, and have a table that associates roles to potential level of sensitive information the user may have on their machine. For example, a division head, or HR employee would likely have relatively high value of sensitive information on their machine.
- Importance of command: For DoD machines, the command that the host is in gives an idea to its contribution to the mission. For example, all other things being equal, a host in 5th fleet is likely more important than a host in the Naval Research Lab. Again, we have a table mapping all the commands to corresponding numeric criticality values ranging from 1 to 5 (Table 2).

- Host operating system: From our domain experts who apply their internal knowledge, we were able to ascertain one of the things they look at is the operating system of the host. Generally, mission critical operations run on windows machines, while Macs are used for general purpose work. We reflect that in a table mapping operating systems to numeric criticality values, ranging from 1 to 5.


The tables we use for each attributes can be tailored to each specific organization with respect to values and mappings. The attributes selected here by themselves are not sufficient to determine asset criticality, but when combined can give us a strong approximation of asset criticality. The attributes are combined using a decision-making algorithm called Technique for Order Preference by Similarity to Ideal Solution (TOPSIS) [cite], which we modified to ensure faster calculations. A detailed description of TOPSIS and our modifications is available in Appendix B.

Table 1. Mapping of services and applications to criticality values

Service/App	Integrity	Availability
File Server	5	3
Web server	3	3
Router	5	5
Firewall	5	5
User machine	1	1
General purpose	1	1
R&D	2	1
Admin machine	5	3
FTP server	5	3

Table 2. Mapping of commands to criticality values

Command	Rating
NRL	1
NavMed (SD)	2
6th fleet	3
SeaLift	3
10th fleet	3
Task force 60	3
7th fleet	4
5th fleet	5



Command Ratings	Meanings
1	low criticality
2	medium low criticality
3	moderately mission critical
4	relatively high criticality
5	extremely mission critical

Step 2. Adjust AC Values for Data Dependencies

Various assets can have data dependency relationships in which assets depend on input from other systems. If a critical asset is highly dependent on input from another (i.e., the provider asset), and cannot complete its mission without this input, then the AC value of the provider asset should be elevated to reflect this. These dependencies are taken into account to tweak AC values higher for assets that have dependent hosts. The number of hosts that depend on a particular asset, the strength of the dependencies, and the AC values of the dependent hosts themselves all factor in contributing to the dependency portion of the asset criticality value.

In summary, our dependency algorithm should possess the following properties:

- Each initial value (i.e., AC value) of the node/assets in the dependency relationship is the independent asset criticality score obtained from the individual TOPSIS calculations.
- The AC value of a provider asset should be a function of its initial AC value, the AC value of the dependent assets, the strength of the dependencies, and the number of assets that depend on it.

The relationship between the provider asset's updated AC value and the dependent nodes' AC values, degree of dependencies or the number of dependent nodes is not linear or exponential. A provider asset that supports 100 other nodes is likely more critical than one that supports 10 nodes, but is not necessarily 10 times more important. The criticality function would likely be defined by a logarithmic relationship between the adjusted criticality value of the provider node and the variables (i.e., AC values of dependent assets, number of dependent assets, and strength of dependencies). Based on this, we take the log base 10 of the sum of all the dependent hosts' AC values multiplied by the strength of their dependencies, and add this value to the original AC value of the provider asset:

$$AC_{new}(p) = AC(p) + \log_{10} (\sum(AC(a_i) \times w_i))$$

where p is the provider asset, $AC(p)$ is the original AC value for p , $AC_{new}(p)$ is the updated value for p , $AC(a_i)$ is the AC value for a dependent asset a_i , and w_i is the strength of the dependency from a_i to p .

Step 3. Adjust the AC Values for User-defined Factors

Particularly in DoD, factors such as 'who is logged on' and 'who owns the asset' may change the criticality of an asset depending on what the value of the factor is at a given time. For example, when a highly ranked official is logged on to a host, this may override the existing (low) AC value of the asset. This is analogous to the Air Force One call sign that is given to any aircraft carrying the President of the United States at that time. Use cases like these are extremely subjective, so we leave it up to the cyber warrior to adjust the values as needed. This can be done within the ACCEPT tool.

Appendix B. Modified TOPSIS Algorithm

First, we explain the original TOPSIS algorithm, and then we discuss the changes we made to it.

Original TOPSIS Method (Technique for Order Preference by Similarity to Ideal Solution)

The TOPSIS approach ranks alternatives according to their closeness to a hypothetical positive ideal solution (zenith) and a hypothetical negative ideal solution (nadir). The domain set of alternatives is defined as a m -dimension Euclidean space. Therefore, each alternative is represented as a point in this space. A basic assumption made here is that each attribute is characterized by monotonic increasing or decreasing utility. Using the TOPSIS principle, the solutions (i.e., most preferred alternatives) are those alternatives which are at the same time farthest from the nadir point and closest to the zenith point, with distance being measured by Euclidean distance. TOPSIS models this principal by defining ‘relative closeness to the ideal solution’, T_j^* according to the following relation:

$$T_j^* = \frac{D_j^-}{D_j^- + D_j^+} \quad (1)$$

Where D_j^- is the m -dimensional Euclidean distance between the j^{th} alternative and the nadir point, and D_j^+ is the m -dimensional Euclidean distance between the j^{th} alternative and the zenith point. According to this method, the alternative A_j is better than A_m if $T_j^* > T_m^*$ or $D_j^- / (D_j^+ + D_j^-) > D_m^- / (D_m^+ + D_m^-)$. This calculation is necessary because there can be many alternatives that are closest to the ideal solution. Also the alternative closest to the ideal solution is not necessarily the farthest from the negative-ideal solution. As can be seen in Figure 14 [20], if we only have alternatives $A_1, A_2,$ and A_3 , they are all equidistance from A^* , the positive-ideal solution, but have different Euclidean distances from A^- . Now, if we consider A_1 and A_4 , A_4 is closer to the ideal solution, but A_1 is farther from the negative-ideal, A^- . The relative distance measurement handles these conflicts.

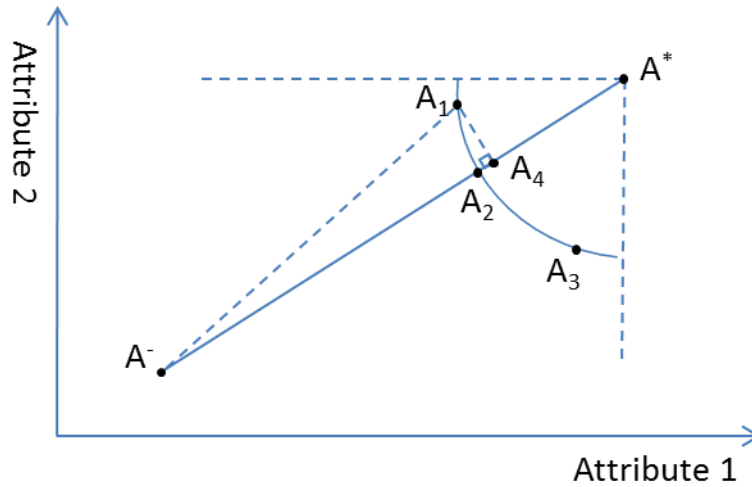


Figure 14. Euclidean distances to positive and negative ideal solutions

In order to calculate the relative closeness to the ideal solution, the normalized decision matrix is first calculated using the vector method:

$$z_{ij} = \frac{x_{ij}}{\sqrt{\sum_{j=1}^n x_{ij}^2}}$$

where x_{ij} is the value of alternative j with respect to attribute i ($1 \leq i \leq m, 1 \leq j \leq n$). Then, the weighted normalized decision matrix is calculated as:

$$v_{ij} = w_i z_{ij}, \quad i = 1, \dots, m, j = 1, \dots, n$$

Where w_i is the weight of the i^{th} attribute. The zenith A^* and nadir A^- are calculated as follows:

$$A^* = \{v_1^*, \dots, v_m^*\} = \{(max_j v_{ij} | i \in I'), (min_j v_{ij} | i \in I'')\},$$

$$A^- = \{v_1^-, \dots, v_m^-\} = \{(min_j v_{ij} | i \in I'), (max_j v_{ij} | i \in I'')\},$$

Where I' is associated with benefit criteria, and I'' is associated with cost criteria¹. Thus, the zenith (positive ideal solution) is made up of the best value of each criterion over all attributes and the nadir (negative ideal solution) is composed of the worst value of each criterion over all the attributes. The respective Euclidean distances are calculated as:

$$D_j^+ = \sqrt{\sum_{i=1}^m (v_{ij} - v_i^*)^2}, \quad j = 1, \dots, n$$

$$D_j^- = \sqrt{\sum_{i=1}^m (v_{ij} - v_i^-)^2}, \quad j = 1, \dots, n$$

The relative closeness is then calculated as in equation (1) to obtain the score for each alternative, after which the alternatives are ranked. An example of applying the original TOPSIS is provided in Table 3. Because we represent asset criticality in terms of requirements for confidentiality, integrity, and availability, a zenith and nadir must be calculated for each. The cells highlighted in yellow are the best value of each criterion over all the attributes, and together compose the zenith. The cells highlighted in purple are the worst value of each criterion and compose the nadir. In other words, the zenith for value of data on the asset for confidentiality is $A^* = \{0.4, 0.25, 0.25, 0.5\}$ and the nadir is $A^- = \{0.4, 0.25, 0.25, 0.3\}$. The distances to the zenith and nadir are computed using these values against each value in the alternatives. For instance, the distance to A^* from Host 1 for confidentiality is

$$D_1^+ = \sqrt{(0.4 - 0.4)^2 + (0.25 - 0.25)^2 + (0.25 - 0.25)^2 + (0.5 - 0.5)^2} = 0$$

and the distance to A^- from Host 1 for confidentiality is

$$D_1^- = \sqrt{(0.4 - 0.4)^2 + (0.25 - 0.25)^2 + (0.25 - 0.25)^2 + (0.5 - 0.3)^2} = 0.2.$$

¹ A benefit attribute is one in which higher measures are more desirable for the decision-making problem, and a cost attribute is one in which lower measures are more desirable. For example, when buying a car, the gas mileage of the car is a benefit attribute, and the price of the car is a cost attribute.

The resulting relative closeness value T_1^* for confidentiality on Host 1 is $0.2/(0.2 + 0) = 1$. Other values are calculated in a similar manner. This way each host has a relative closeness score composed of the three components of confidentiality, integrity and availability. For instance, Host 2 has a score of $\{0.0359, 0\}$. For ease of comparison, and also to have a one-dimensional score, we take the vector of these three values and let that be the asset criticality value for that host. In this manner, host 1 is the most critical asset with a (vector) AC value of 0.696, and host 2 is the least critical asset with an AC value of 0.207. We can easily multiply these values by 10 to provide a range of values from 0 to 10.

Table 3. Sample calculation of asset criticality using original TOPSIS

FACTORS	W	Host 1			Host 2			Host 3		
		C	I	A	C	I	A	C	I	A
Criticality level of apps (Raises I, A) + Network (C)	0.4	1	2	3	1	1	3	1	1	5
Value of data on the asset (raises C)	0.25	1	0	0	1	0	0	1	0	0
Importance of command	0.25	1	1	1	1	2	1	1	1	1
Operating System (M, L, W)	0.1	5	5	5	3	3	3	5	5	5
Distance to Zenith		0	0.25	0.8	0.2	0.447	0.825	0.731	0.687	2.077
Distance to Nadir		0.2	0.447	0.2	0	0.25	0	0.731	0.687	2.077
Relative Closeness		1	0.641	0.2	0	0.359	0	0.5	0.5	0.5
Vector		0.696			0.207			0.5		

Modified TOPSIS Method

TOPSIS calculates its values across all other alternatives using the concept of positive and negative ideals. In particular, the positive ideal and negative ideal in TOPSIS are relative values, calculated using the given values in the decision matrix. The use of relative values for positive and negative ideals is particularly problematic for our dynamic environment where asset scores can change, assets can be added or removed, and assets need to be compared across different commands. When a change in an asset's value affects the positive or negative ideal, all assets' distances to the positive and/or negative ideal (and thus the relative closeness) need to be recalculated. For example, going back to the example given in Table 3, assume that 'Value of data on the asset' for confidentiality changed from 1 to 3 for Host 1. The resulting positive ideal for confidentiality would become $A^* = \{0.4, 0.75, 0.25, 0.5\}$. The negative ideal has not changed, but the positive ideal has, requiring recalculation of all assets' distances to the positive ideal as well as their relative closeness values. When we are dealing with 100s of thousands of assets, such as in a real cyber environment, frequent changes like these will require continuous recalculations and slow down our performance.

The problem with using TOPSIS in our environment stems from the way positive and negative ideal values are selected. To avoid this problem, we modify the distance measurement calculation so that *absolute* positive and negative ideal values are used as the new zenith and nadir, respectively. In other words, we create a hypothetical asset that has all possible maximum values and a hypothetical asset that has all possible minimum values, and calculate the distance to the zenith and nadir from these values respectively. For example, knowing that all our values range from 0 to 5, the new absolute zenith is $A^* = \{2, 1.25, 1.25, 0.5\}$ and the new absolute nadir is $A^- = \{0, 0, 0, 0\}$ after weights have been applied.

Using these values to calculate distances and then relative closeness measurements, the new TOPSIS values would be that of Table 4.

Table 4. Calculation of asset criticality using modified TOPSIS

FACTORS	W	Host 1			Host 2			Host 3		
		C	I	A	C	I	A	C	I	A
Criticality level of apps (Raises I, A) + Network (C)	0.4	1	2	3	1	1	3	1	1	5
Value of data on the asset (raises C)	0.25	1	0	0	1	0	0	1	0	0
Importance of command	0.25	1	1	1	1	2	1	1	1	1
Operating System (M, L, W)	0.1	5	5	5	3	3	3	5	5	5
Distance to Zenith		2.135	1.562	1.281	2.145	1.778	1.296	0.731	0.687	2.077
Distance to Nadir		0.731	0.976	1.324	0.612	0.707	1.262	0.731	0.687	2.077
Relative Closeness		0.255	0.385	0.508	0.222	0.284	0.493	0.5	0.5	0.5
Vector		0.396			0.353			0.5		

Unlike the original TOPSIS approach that uses only the values available in the decision matrix, these zenith and nadir values are fixed. The advantage of this is that if an asset's value changes or the asset itself is removed from the decision-making process, the zenith and nadir values do not change, and all the sub-calculations for relative closeness do not need to be recalculated for each asset: if an asset's value changes, then only that asset's relative closeness (to the absolute positive and negative) needs to be recalculated. If an asset is removed, no recalculation of any relative closeness values is required since the zenith and the nadir are always computed using absolute values. Modifying TOPSIS in this manner allows asset criticality values to be comparable and minimizes recalculation time. In a dynamic environment, this type of approach is advantageous and necessary.

Appendix C. Host Exposure Module

The purpose of the Event Priority project is to prioritize incoming network events based on their severity as well as the criticality of the hosts that they target and other pertinent information. The overall process is modeled as a framework, composed of several modules that together enable more accurate prioritization of events.

Overview

One of the more significant factor of the EP framework is the Host Exposure factor that calculates a measure of risk (numeric score between 0 and 10, with 10 being extreme risk) of a host's exposure to potential attacks. This assists users in determining the likelihood of a host being the target of an attack, of an attack being successful, and the potential damage these attacks can cause. Attacks are attempts by an adversary to cause harm to valuable assets, by trying to exploit one or more vulnerabilities (weaknesses or flaws in the system). The number of vulnerabilities a host has and the potential level of damage these vulnerabilities possess determine the level of threat (i.e., exposure) to the system.

The Host Exposure factor evaluates all of the vulnerabilities on a given asset/host to calculate a proper measurement. This measure takes into account the number of vulnerabilities (attack surface) on an asset and their damage potential either collectively (max impact) or individually (max vulnerability). The input to this module is the list of vulnerabilities in Common Vulnerability Scoring System (CVSS) format, which can be obtained from a vulnerability scanner such as Nessus. The CVSS provides a standardized method for rating known vulnerabilities, and has been adopted by many security vendors to describe and rank vulnerabilities.

Inputs to the Module

CVSS provides severity scores for vulnerabilities by combining many different factors in a numeric equation. However, we could not use the base scores provided as-is in our calculations. Using the CVSS-provided base scores by themselves does not present an accurate measure of a host's potential exposure for one main reason: the CVSS base score formulas were designed to compare two different vulnerabilities, not two different hosts with multiple vulnerabilities. As a result, many of the factors used to determine base scores become obfuscated by the CVSS equations. It is then difficult to rank the exposure of a system by using some combination of CVSS base scores. Without breaking a CVE score into its base components, there is no straightforward way to determine if a combination of specific vulnerabilities will result in a complete compromise to the host system. In order to best evaluate a host, the framework should take into account how a set of vulnerabilities can be combined in order to completely compromise the host instead of simply using the base score.

Even though the CVSS base score is not used, its base components that describe the vulnerability, such as Confidentiality, Integrity, and Availability (CIA) impact are useful in determining a host's overall exposure. For Host Exposure, the impact to confidentiality, integrity and availability, the access vector, complexity and the authentication requirements (all provided from the CVSS base scores) will be used to calculate potential exposure. The base components and their definitions are provided here:

- Confidentiality Impact – Measures the impact on confidentiality of a successfully exploited vulnerability (Complete, Partial, None)
- Integrity Impact – Measures the impact on integrity of a successfully exploited vulnerability (Complete, Partial, None)
- Availability Impact – Measures the impact on availability of a successfully exploited vulnerability (Complete, Partial, None)
- Access Vector - Reflects how the vulnerability is exploited (Local, Adjacent Network, Network); the more remote an attacker can be, the greater the vulnerability score for this factor
- Access Complexity – Measures how difficult the vulnerability is to exploit (Low, Medium, High)
- Authentication The number of times an attacker must authenticate before launching the attack (Multiple, Single, None); the fewer authentication instances required, the higher the vulnerability score of this factor

For Host Exposure, we translate these qualitative values into quantitative ones. An initial value of 1 indicates a complete loss of a confidentiality, integrity, or availability requirement for a system. Partial loss is assigned a 0.3 (in other words, more than 3 partial losses are equivalent to a complete loss), and no loss is assigned a value of 0. Impacts to the CIA requirements can be expressed in a vector format which each vector component $\langle C, I, A \rangle$ represents the loss for that component. For example, $\langle \text{Complete}, \text{Complete}, \text{Partial} \rangle$ represents a complete loss of confidentiality, a complete loss of integrity, and a partial loss of availability. For the access vector, exploitation complexity and authentication components, 1 is used to denote a remotely exploitable vulnerability, no authentication and low complexity. The values used to describe the other levels can be flexible, but should all be less than 1.

Calculations

Calculating host exposure consists of combining several different factors based upon the vulnerabilities obtained from a network or host scanner. These factors take into account the number of vulnerabilities required to compromise a system, how severe those vulnerabilities are, and how large of an attack surface the host has. The factors are: the initial compromise level, max impact level, max vulnerability, attack surface and the max access, access complexity and authentication requirements. Instead of evaluating the access vector, access complexity and authentication with respect to the corresponding vulnerability, they are evaluated with respect to the overall system. This is due to the fact a remote access exploit can in turn allow local vulnerabilities to be exploited. To separate the individual Host Exposure values across the three dimensions of confidentiality, integrity, and availability, these factors are calculated separately across the three dimensions. The final combined value will be the vector of the three separate host exposure values.

**** Note:** V is used to denote the entire set of vulnerabilities for a host h . v denotes a single vulnerability within the set of vulnerabilities for the host. v_C is the confidentiality impact, v_I is the integrity impact and v_A is the availability impact for vulnerability v .

- **Initial Compromise level:** $IC(V)$ - This represents the host's potential level of compromise based upon the known vulnerabilities found by the scanner. To calculate the initial compromise level, $IC(V)$, we determine the sum of each of the impact measures (confidentiality, integrity and availability) and take the minimum of 1 and that sum (cells B12, C12, and D12 in Table 5):

$$IC_{C,I,A}(h) = \min(1, \sum v_{C,I,A}), \forall v \in V$$

where $v_{C,I,A}$ is the vulnerability's impact to confidentiality, integrity, or availability, respectfully.

These values tell us whether or not there is a way to completely compromise the system by exploiting some combination of the existing vulnerabilities. For example, a value of $IC_I = 0.3$ means that even though we summed all the vulnerabilities' integrity impacts, we could not obtain a complete loss of integrity. On the other hand, if we have a value of $IC_C = 1$, that means that some combination of vulnerabilities in the system provides complete loss of confidentiality. However, these values do not tell us the number of vulnerabilities that would be required for a successful compromise.

- **Max Impact:** $MI_{C,I,A}(h)$ – The max impact level represents the potential maximum impact of a vulnerability existing on host h to each of the corresponding CIA measures. This value determines the upper bound on the number of vulnerabilities required in order to completely compromise a host for a given dimension. It is obtained by calculating the maximum impact value for C, I, and A of all the vulnerabilities (cells B13, C13, and D13 in Table 5):

$$MI_{C,I,A}(h) = \max(v_{C,I,A}), \forall v \in V$$

In a system with a maximum impact of $MI_A(h) = 0.3$ an attacker using the most effective vulnerabilities would need to at most use 4 partial vulnerabilities to completely compromise the availability of the system. This is an upper bound because when the three values are combined, we do not know if the same vulnerability can be used to attack the confidentiality, integrity, and availability. For example, if we have $(MI_C(h), MI_I(h), MI_A(h)) = (1, 1, 0.3)$ then an attacker using the most effective vulnerabilities would need to at most use 6 vulnerabilities to completely compromise the system; 2 vulnerabilities would be required to compromise confidentiality and integrity (1 each), and at least 4 partial vulnerabilities to equal a complete compromise of availability.

- **Max Vulnerability:** $MV(h)$ – The max vulnerability represents the single greatest vulnerability in the set of known vulnerabilities for that host. This value determines the lower bound on the number of vulnerabilities required to compromise a system across all three dimensions, and tells us how much damage one vulnerability can cause on the host when exploited. For this reason, this value is not determined separately for confidentiality, integrity, or availability. This value is calculated using the vector magnitude approach and normalized to 1. In Table 5, we calculate the vector magnitude of each vulnerability (in Column E) and choose the maximum value from this column as the Max Vulnerability. A maximum of 1 indicates that a single vulnerability can completely compromise the system across all three dimensions. Midrange values represent systems where 2 or more vulnerabilities must be used (at least) to completely compromise the host.

$$MV_{C,I,A}(h) = \max\left(\sqrt{(v_C)^2 + (v_I)^2 + (v_A)^2}\right), \forall v \in V$$

- **Max Access Vector:** $AV(h)$ - This value represents the minimum access an attacker would need in order to compromise the system. It is obtained by taking the maximum of the access vectors (Column F, Table 5), v_{access} , of all the vulnerabilities in the system. For example, an attacker could exploit local vulnerabilities after first using a remote exploit to access the system.

$$AV(h) = \max(v_{access}), \forall v \in V$$

- **Average Access Complexity:** $CA(Vh)$ – This value represents the average level of complexity required to attack a system, obtained by taking the average of the complexity requirements (Column G, Table 5), v_{comp} , for all the vulnerabilities on the host.

$$CA(h) = \text{avg}(v_{comp}), \forall v \in V$$

- **Max Authentication:** $AN(h)$ – As with the max access vector, the max authentication represents the minimum number of authentications a hacker would require when attacking the system using one or more of the vulnerabilities. It is obtained by taking the maximum of the authentication requirements (Column H, Table 5) for all the vulnerabilities on the host.

$$AN(h) = \max(v_{auth}), \forall v \in V$$

- **Combined Access/Complexity/Authentication:** $ACA(h)$ – The max access vector, complexity and authentication could either be expressed individually, or as a single vector by using the magnitude of these three factors (cell J5 in Table 5).

$$ACA(h) = \sqrt{\max(v_{access})^2 + \text{avg}(v_{comp})^2 + \max(v_{auth})^2}, \forall v \in V$$

- **Attack Surface:** $ATT(h, x, y)$ – The last major factor to consider when analyzing the vulnerabilities in the system is the attack surface. For example, a complete, remote and low complexity compromise on two separate hosts will both result in a final Host Exposure score equal to 10 even if the second host has 5 times the vulnerabilities of the first host. Both hosts are severely exposed, but the second host can be considered to be the more vulnerable system simply because it has a larger attack surface. In order to account for this type of situation, a measure of the attack surface can be added to the score independent of the other calculations. In addition, it should follow a logarithmic scale until it maximizes the allowed range for the calculation.

$$ATT_{C,I,A}(h, x, y) = \max(\log_x \left(\text{avg} \left(\sum v_{C,I,A} \right) \right), y), \forall v \in V$$

Where x is the base for the logarithmic function and y is the limit of the function. This function gives an idea of how many vulnerabilities are on the host, and the potential degree of compromise they possess. Modifying the base of the log function determines how much of an attack surface is needed to raise the final value. For example, using a base of 2 would require at least 4 additional

complete compromises for the base value to be modified. Similarly using a base of 3 would require 9 complete compromises.

Final Score

Calculating the final Host Exposure $HE(h_i)$ value for host h_i is fairly straightforward. The final value should be on a scale of 0 to 10 and should identify a host's relative potential exposure to attack. Starting with the Initial Compromise level and normalizing it to 10, the final value decreases if any of the other factors have a value less than 1. In a worst-case scenario where a single vulnerability completely compromises the system with low complexity, remote access and no authentication, the resulting value will be a 10, indicating severe exposure.

$$HE_{C,I,A}(h_i) = IC_{C,I,A}(h_i) * (10 - y) * MI_{C,I,A}(h_i)^{W_{MI}} * MV(h_i)^{W_{MV}} * ACA(h_i)^{W_{ACA}} + ATT_{C,I,A}(h_i, x, y)$$

Where W_{MI} is the weight of the max input, W_{MV} is the weight of the max vulnerability, and W_{ACA} is the weight of the combined access vector, access complexity and authentication vector.

Weighting

Since each factor used to modify the final score ranges from 0 to 1, weighting that specific factor without affecting the rest of the function is accomplished using multiple powers. Multiplying by another constant is not useful since that constant would affect the entire function, and the weighted factor could no longer be contained within the 0 to 1 range. Using a power to weigh a factor allows for each factor to be given individual weights. Weights are also assigned in the 0 to 1 range, where 0 represents no affect on the final score ($w^0 = 1$) and 1 represents full effect ($w^1 = w$). This value is the relative rate of each factor on the final score. So, a weight of 0.5 is half as effective as a weight of 1. In order to contain the final score in the range 0 to 10, the normalizing factor of $PE(V)$ should be $10 - y$, where y is the max of the attack surface function $ATT(V, x, y)$.

Table 5 shows how the various components are calculated and combined to obtain the Host Exposure value for one host with eight vulnerabilities on it. The vulnerability IDs are masked by using 'CVE-NRL' as the prefix.

Table 5. Host Exposure calculation worksheet

	A	B	C	D	E	F	G	H	I	J	K
1		Conf_Imp	Int_Imp	Ava_Imp	Norm(Mag)	Access	Complexity	Auth			Weight
2	Vulnerabilites	0.00	0.00	0.30	0.17	0.80	0.80	0.80	Initial Value	7.51	0.50
3	CVE-NRL-0002	0.30	0.30	0.00	0.24	0.80	0.80	0.80	Max Impact	0.83	1.00
4	CVE-NRL-0003	1.00	1.00	0.00	0.82	0.80	0.80	0.80	Max Vulnerability	0.83	0.50
5	CVE-NRL-0004	1.00	1.00	0.00	0.82	0.80	0.80	0.80	Access/Complexity/Auth	0.80	1.00
6	CVE-NRL-0005	1.00	1.00	0.00	0.82	0.80	0.80	0.80	Unweighted Value	4.58	
7	CVE-NRL-0006	1.00	1.00	0.30	0.83	0.80	0.80	0.80	Attack Surface	1.25	
8	CVE-NRL-0007	1.00	1.00	0.00	0.82	0.80	0.80	0.80	Final Value	4.91	
9	CVE-NRL-0008	0.30	0.00	0.00	0.17	0.80	0.80	0.80			
10	CVE-NRL-0009	0.30	0.00	0.00	0.17	0.80	0.80	0.80			
11	SUM	5.90	5.30	0.60	Max -->	0.80	0.80	0.80			
12	Init. Compromise	1.00	1.00	0.60	0.75						
13	Max Impact	1.00	1.00	0.30	0.83						

