Annex C – EXPERIMENTATION ENVIRONMENT AND TOOLS

Niranjan Suri and Kelvin Marcus
U.S. Army Research Laboratory (ARL)
UNITED STATES

Levent Mīsīrlīoğlu
MilSOFT Yazilim Teknolojileri A.Ş
TURKEY

Markus Peuhkuri
Aalto University
FINLAND

Maggie Breedy
Florida Institute for Human & Machine Cognition (IHMC)
UNITED STATES

This annex provides an overview of the Anglova experimentation environment and surrounding tools that were created or co-opted for the activities of the NATO IST-124 RTG. The Anglova military scenario itself is described in Annex A: "Operational Perspective for IST-124" and details about the mobility patterns and experimentation methodologies for the Anglova scenario are described in Annex B: "Emulation-based Experimentation and the Anglova Scenario". The IST-124 RTG instantiated the Anglova scenario using the Extendable Mobile Ad-hoc Networking Environment (EMANE). This annex provides some detail on EMANE and discusses alternate deployment models for the experimentation environment. In particular, two deployment models are discussed – the US Army Research Laboratory's (ARL's) Dynamically Allocated Virtual Clustering (DAVC) and VMware ESXi. Additional details about using DAVC to conduct experiments are covered in Annex D: "IST-124 Experimentation Execution".

This annex is organized as follows:

- Section C.1 provides some background on EMANE.
- Section C.2 provides an overview of deploying Anglova over the DAVC environment.
- Section C.3 provides an overview of deploying Anglova over the VMware ESXi environment, including step-by-step directions to configuring the networks and Virtual Machines (VMs) within ESXi.
- Section C.4 discusses two different visualization tools – SDT3D and Mirage, along with the OpenStreetMap dataset that has been created to support the Anglova scenario.
- Section C.5 discusses tools for playing the Anglova scenario and driving the emulation.
- Section C.6 discusses tools for generation of custom pathloss data.
- Section C.7 discusses lessons learned.
- Section C.8 describes some open issues that are yet to be resolved.

C.1 EMANE OVERVIEW

The Extendable Mobile Ad-hoc Networking Emulator (EMANE) [1] was developed by CenGen (now AdjacentLink) under sponsorship by the US Naval Research Laboratory (NRL), the Office of the Secretary of Defense (OSD), and the US Army Research Laboratory (ARL). It is freely available on GitHub and provides a flexible framework for emulating multiple types of radios. EMANE primarily addresses the Physical (PHY) and Media Access (MAC) layers of the network stack and includes three different radio models – a generic RF Pipe model, an 802.11abg model, and a TDMA model. Each of these models offers various customizations of the radio parameters, antennas, and error models. While models of some specific tactical radios exist, they are not available in the public domain and hence will not be discussed in this document.

EMANE supports a flexible deployment model – centralized, fully distributed, or a hybrid of the two. Each radio interface of a network node in the emulated network is represented within EMANE by an instance of a Network Emulation Module (NEM). The NEMs instantiate the PHY and MAC layers of the radio being emulated and each EMANE instance can contain one or more NEMs. In the centralized model, all the NEMs are instantiated with a single instance of EMANE. In the fully distributed model, each EMANE instance contains only one NEM. In the hybrid model, there are multiple instances of EMANE each with multiple NEMs. In the hybrid and distributed deployment models, the NEMs communicate with each other using an Over The Air (OTA) channel, which uses UDP multicast over a control network (which is independent of the emulated network interfaces). The NEMs also react to control events that are sent to them over the EMANE event channel, typically also via UDP multicast. For example, when playing the Anglova scenario, the position and pathloss updates are sent to EMANE via this control channel. See the EMANE documentation for more details [1].

C.2 DYNAMICALLY ALLOCATED VIRTUAL CLUSTERING (DAVC) DEPLOYMENT

The NATO IST-124 RTG uses the US Army Research Laboratory’s Dynamically Allocated Virtual Clustering (DAVC) Management System to deploy the Anglova scenario using the distributed EMANE emulation model. In this emulation model the EMANE software is installed within VMs that also execute the applications that are the subject of the experimentation (routing protocols, data dissemination protocols, etc.) and whose performance is being evaluated.

DAVC is a web-based virtualization service and cloud-operating environment that creates complex virtual experimentation clusters that can be used for simulation-based, emulation-based, and hybrid field/emulation experimentation [2]. DAVC deploys networked clusters composed of VMs tailored to user specifications. The DAVC management system abstracts away test-bed infrastructure configuration through automated provisioning processes that configures the virtual networking for each VM [2]. Clusters created by DAVC are heterogeneous, so each VM can have different OSs, application sets, and hardware attributes such as RAM, CPU cores, hard disk, and network interfaces. DAVC users can register custom VMs as templates that can be used within their experimentation clusters.

The NATO working group created a custom Linux Ubuntu 16.04 VM to represent a single Anglova scenario node. This template VM was preinstalled with the applications necessary for running the Anglova scenario including EMANE, the Multi Generator (MGEN) traffic generator [3], and the OLSRv1 and OLSRv2 routing protocols [4]. The VM was preinstalled with the various EMANE radio models, mobility and pathloss configuration files specific to the Anglova scenario. Custom scripting to bootstrap the Anglova scenario and emulation environment was also preinstalled within the VM. Once registered with DAVC, the VM is then used as a template within a DAVC experimentation cluster to run the Anglova scenario.

Using DAVC, the Anglova scenario is distributed with a 1-to-1 mapping with each Anglova node running within a single DAVC virtual cluster node. The entire 283 node scenario can run within a 284 node DAVC cluster as shown in Figure C-1. When deployed in this manner node 284 acts as the experimentation orchestration node and is responsible for executing the bootstrap scripting that launches the various applications and EMANE on the remaining 283 nodes. Also shown in Figure C-1 are the two networks DAVC auto-configures for the 284 node cluster. The first network is an out-of-band control/debug network that allows node 284 to communicate with and execute experimentation instructions on the other 283 nodes. The second network is the experimentation network that EMANE uses to overlay the emulated Anglova network channel upon – the OTA channel.

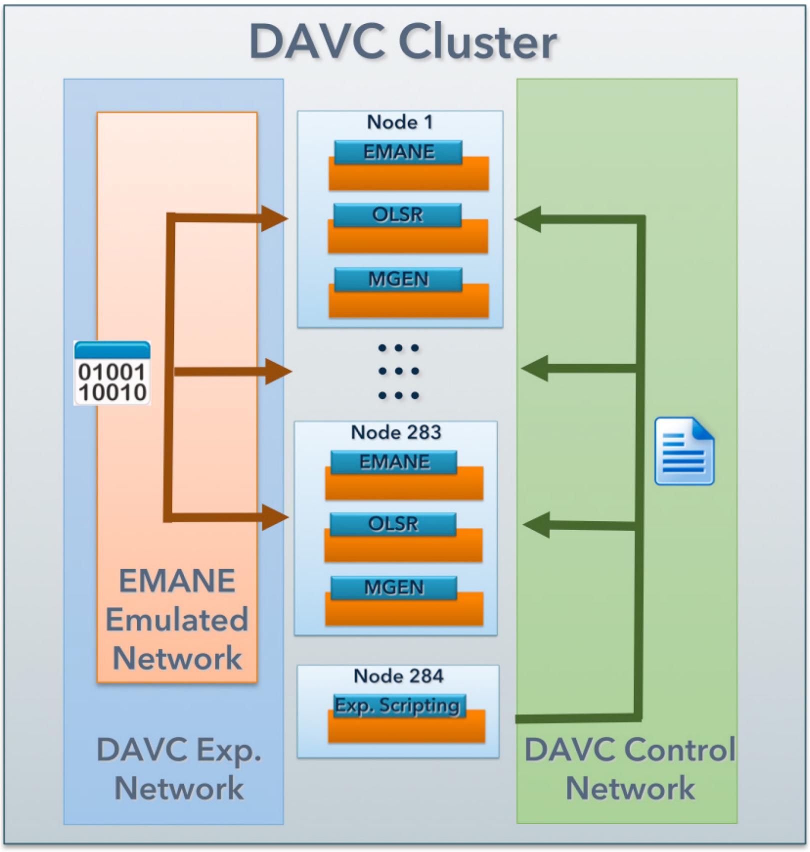

Figure C-1: The 283 Node DAVC Cluster and Networks. The emulated EMANE network is overlaid on the DAVC experimentation network while experimentation commands flow across the DAVC control network.

The emulation environment provides the flexibility to deploy subsections of the entire 283 node scenario within a DAVC cluster. If a researcher is only interested in experimenting with the 159 mobile nodes contained in Vignette 2, the bootstrap scripting can map only those nodes contained within that portion of the Anglova scenario to 159 DAVC cluster nodes, where the corresponding EMANE configuration files will be run. This type of deployment model that uses only a subset of the Anglova scenario was used in the experiment performed in Ref. [5]. The authors were interested in evaluating the scalability of the OLSRv2 routing protocol on company and multi-company sized topologies ranging from 24 nodes to 96 nodes, and hence deployed several 96 node DAVC clusters and configured the experimentation scripting to only run a portion of the scenario that contained four 24 node companies.

The DAVC model also supports experimentation concurrency, which allows multiple experiments to be executed in parallel. The experiments performed in Ref. [5] are an example of this model. Multiple DAVC clusters were deployed where each hosted a different experimental test case. Some experiments were run in parallel where each executed a different version of the OLSR routing protocol and different parameters were provided to the experimentation scripting that varied the MGEN background traffic generation model, network traffic shaping, and the number of Anglova nodes involved in the experiment. This feature of the DAVC model has the effect of reducing overall experimentation runtime.

C.3 VMWARE ESXI DEPLOYMENT

In addition to the DAVC environment, the Anglova scenario was also deployed using a second, alternate configuration using the VMware ESXi virtualization platform. This deployment uses the hybrid model for

EMANE configuration, where n virtual machines have one EMANE Server VM that runs all of the emulation components for those n VMs. An example configuration is shown in Figure C-2. Note that the test VMs are designated Test Node-n (TN-n) and there could be multiple TNs per physical server, with a recommendation of one VM per CPU core. Each test VM is also configured with at least two network interfaces, one for the application data and one for control traffic (similar to the DAVC configuration). Each physical server also contains an EMANE Server VM, which runs all of the EMANE components. One advantage of this deployment model is that the test VMs do not have to run any of the EMANE components. In fact, the network link to the test VM could also be connected to other computing and network devices (e.g., an embedded device, proprietary hardware, radio, a WiFi access point), which allows hybrid experimentation with VMs and other physical hardware in the loop.

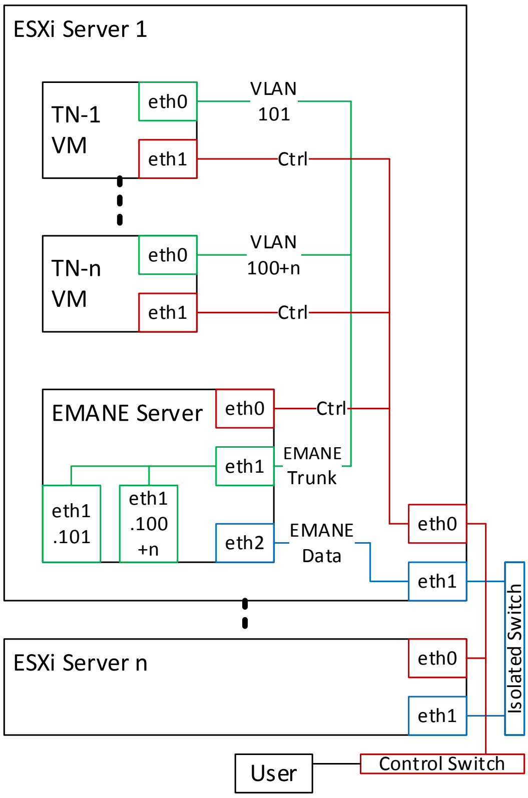

Figure C-2: Anglova Deployment Over ESXi.

The network shown in green is configured as a virtual switch within ESXi, with each test VM connecting one or more interfaces to this virtual switch. Each connection is assigned a different VLAN ID. The EMANE Server VM connects to this switch using a VLAN trunk, which uses the 802.1q protocol. Within the EMANE Server VM, multiple virtual interfaces are created, one per VLAN, which are then managed by EMANE. This configuration results in any traffic generated in the data interfaces of the TNs to be transferred to the corresponding virtual Ethernet interfaces inside the EMANE Server, where each interface is mapped to one NEM. At runtime, EMANE reads from each interface, applies the necessary communications effects, and writes the data out to the interfaces that should receive the data. The network shown in blue is for EMANE data (OTA and events) exchanged between the different physical servers. Finally, the network shown in red is the control network used by a user to log into the VMs, start/stop the experiments, and collect data.

The basic principle of this deployment model is to ensure that any data generated by one TN VM is isolated within a VLAN and is not directly seen by any other TN VM directly. Instead, the data is sent to the EMANE

Server VM, where the data will be consumed by EMANE. Once EMANE decides upon which other TN VMs that should receive that data, it is retransmitted on the appropriate VLAN(s) back to the TN VMs.

The following steps detail the process of configuring one or more ESXi servers to support this deployment model for Anglova. This document assumes that Version 6.5 of the ESXi server is utilized. The management screens shown are using the VSphere Client, and not the web-based interface (although it should be possible to use the web-based interface as well).

After a fresh install of ESXi, the VSphere Client should show something along the lines of Figure C-3. This particular server has 32 logical processors, so it could host up to 32 nodes in the Anglova scenario. Additional nodes could be deployed on additional servers (details later).

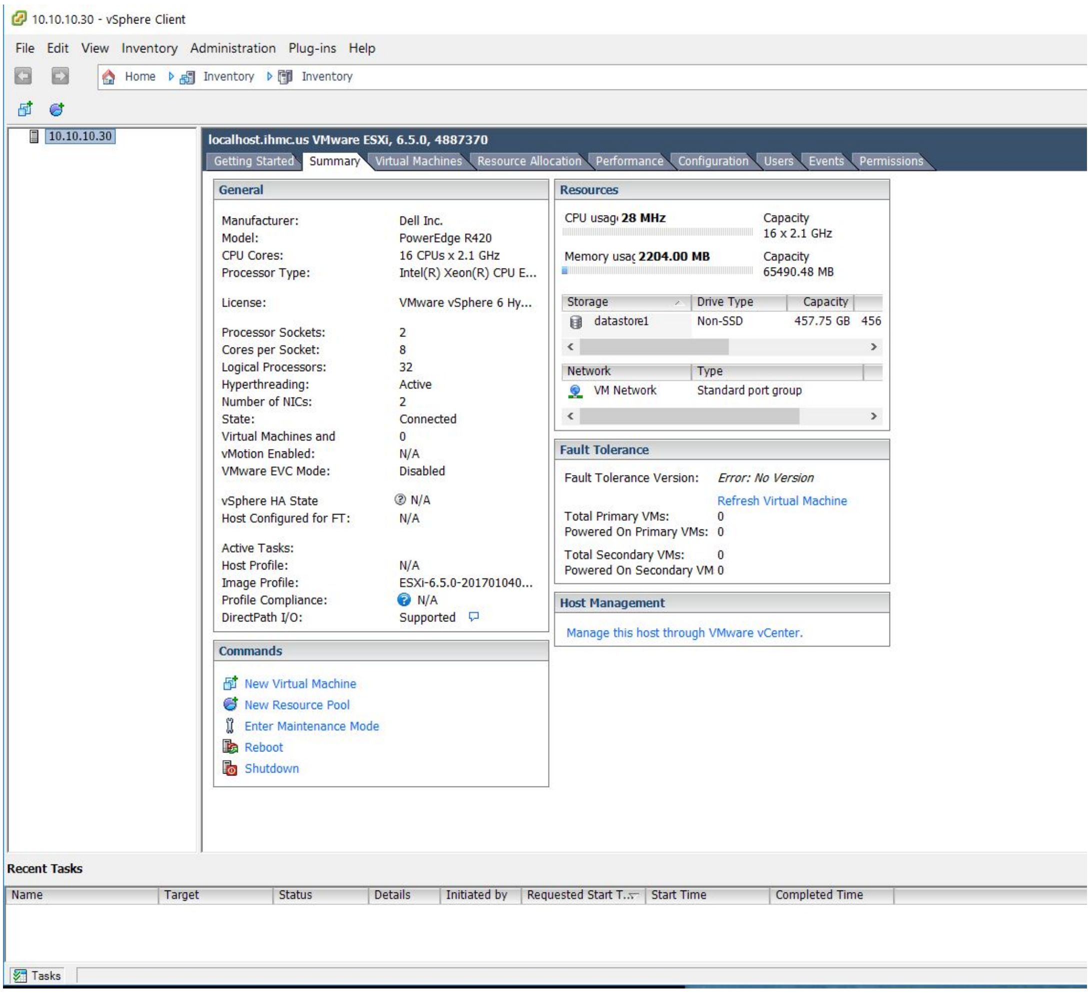

Figure C-3: VSphere Client View After Installation of ESXi Server.

An important part of configuring the ESXi server is setting up the virtual switches and VLANs within the server in order to replicate a topology like the one shown in Figure C-2. Within the VSphere Client, this is accomplished by first selecting the server (10.10.10.30 in this case), clicking on the Configuration tab, and then selecting the Networking option. The Add Networking option is then used to create a new Virtual Switch, of type vSphere Standard Switch, as shown in Figure C-4.

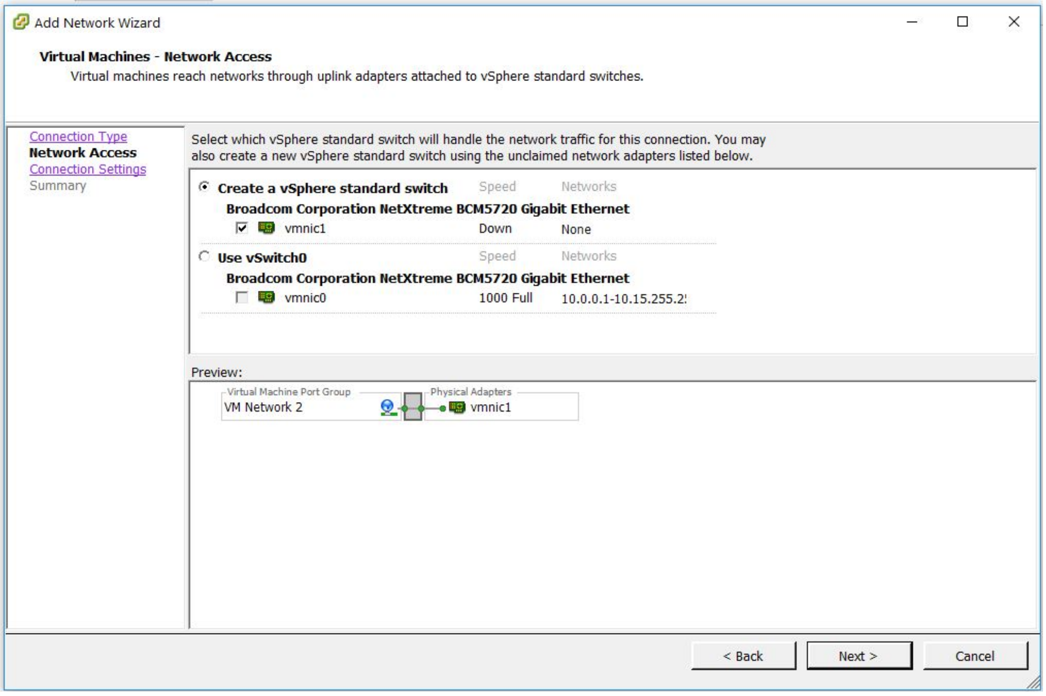

Figure C-4: Creating a vSphere Standard Switch Within ESXi.

Two virtual switches need to be created; the first one for all the VLANs for each VM and the VLAN trunk – called EMANETrunk, and the second one for EMANEData, which allows a multi-server EMANE installation. The EMANETrunk is a trunk for all of the VLANs used by the different TN VMs to transfer data to/from the EMANEServer VM. As shown in Figure C-5, the VLAN ID for this virtual switch should be set to ALL. Note that this particular virtual switch does not need to be connected to a physical Ethernet network, as it only needs to be connected to VMs internal to the server.

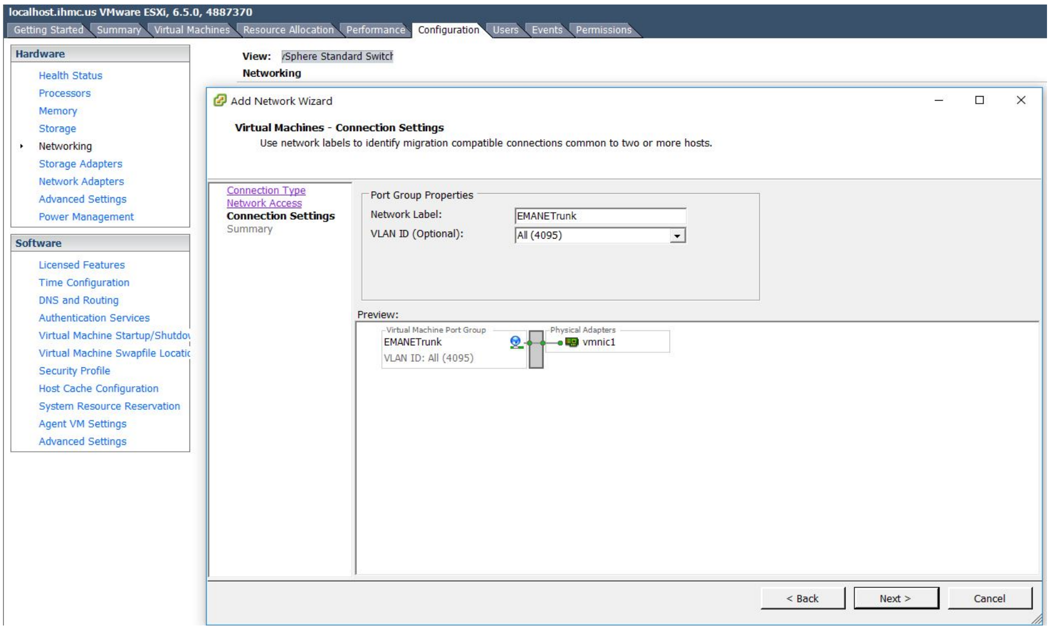

Figure C-5: Configuring the VLAN for the EMANETrunk Virtual Switch.

It is important that the security properties of the EMANETrunk switch are configured to allow Promiscuous Mode. This configuration is set in the Security tab of the EMANETrunk Properties dialog and is shown in Figure C-6.

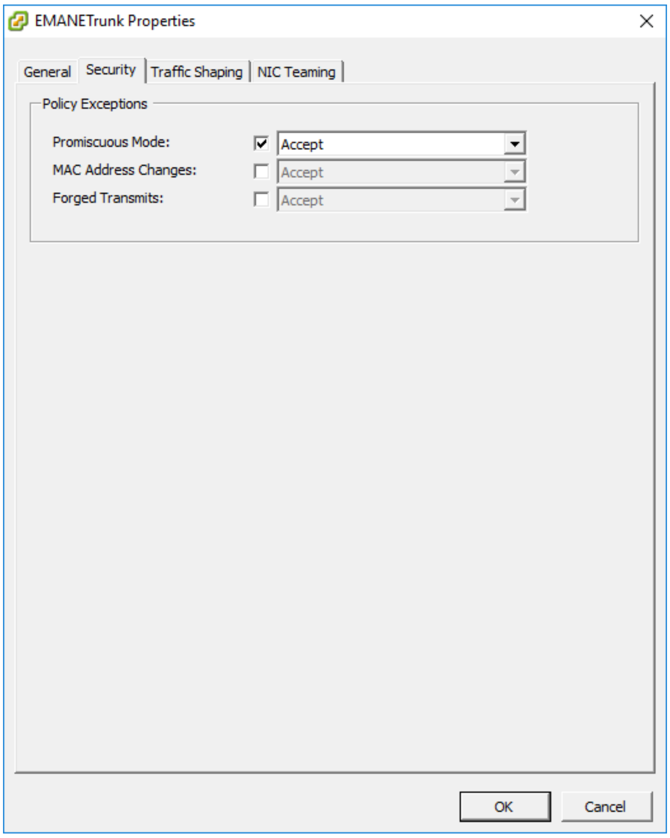

Figure C-6: Configuring Security Properties for EMANETrunk.

The next step is to add VLANs for each TN VM to the same virtual switch. In this example, the VLANs are numbered starting at 100 and in increments of 10 for each test VM. This provides room to add additional VLANs later in case a VM needs multiple interfaces (as some do within the Anglova scenario). Figure C-7 shows the configuration screen for ESXi after deploying EMANE within a VM named EMANEServer and twelve VMs that represent the first twelve nodes of Company 1 within the Anglova scenario. Note that all the VLANs show up as interfaces that can be assigned to specific VMs being deployed on the server.

Another vSphere virtual switch needs to be created for EMANEData. This is only required if the test VMs will span multiple physical servers, and there will be multiple EMANEServer VMs to handle all of the VMs. The EMANEData network needs to be connected to a physical network (as shown in Figure C-2), so that EMANE can pass the Over The Air (OTA) traffic between the multiple EMANE VMs on each physical server. Creation of this virtual switch is shown in Figure C-8.

Each of the deployed test VMs need to have two network interfaces. The first interface must be connected to a unique VLAN, which will guarantee that its network traffic will only be sent to the corresponding NEM in the EMANEServer. The second, control interface, is connected to the VM Network, which is typically used to be able to log into the VM and otherwise communicate with it. Applications that are being evaluated need to be configured to transmit their traffic on the test VLAN interfaces and not on the control interface. An example of this configuration for the first node in the first company in the Anglova scenario is shown in Figure C-9.

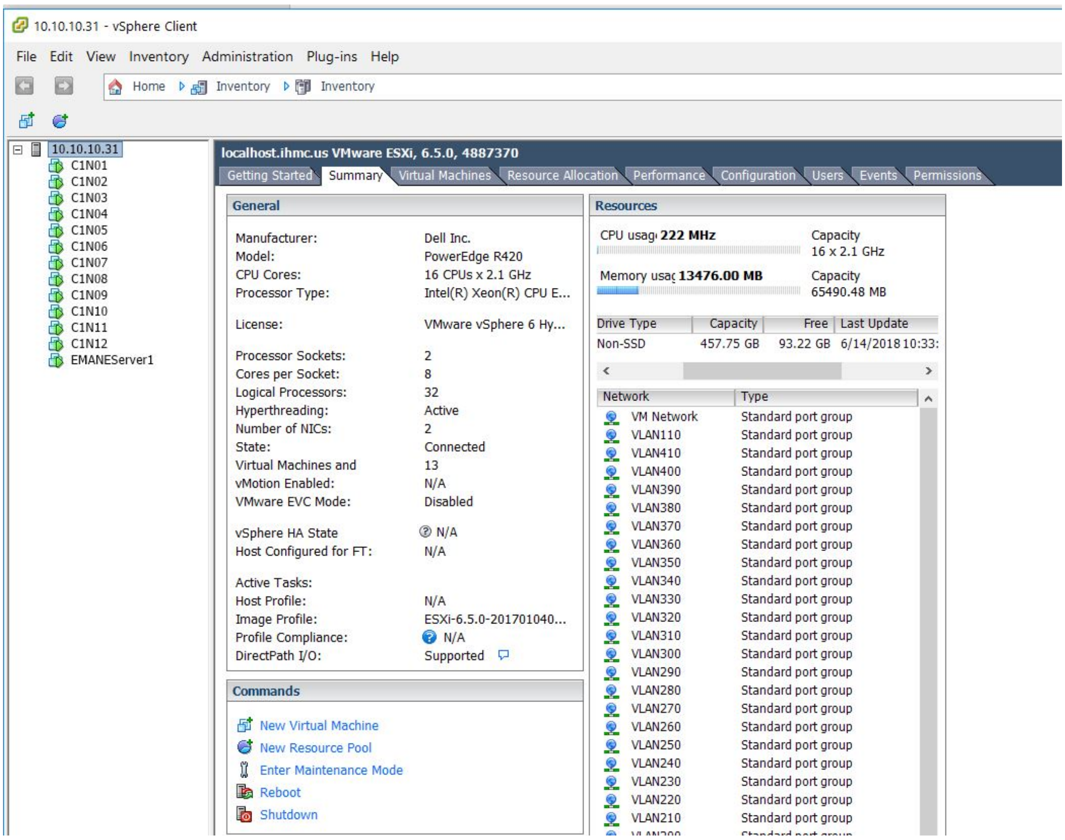

Figure C-7: Status of ESXi Server After Installation of EMANEServer VM and 12 Nodes.

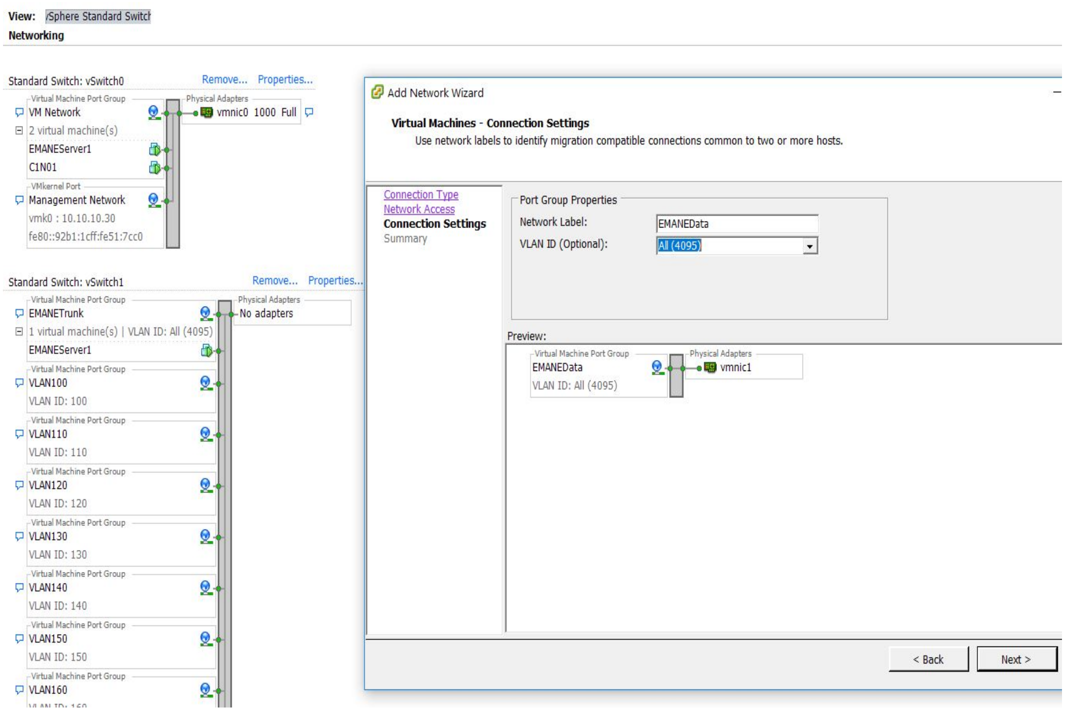

Figure C-8: Creating Another vSphere Switch and Network for the EMANE Over The Air (OTA) Traffic.

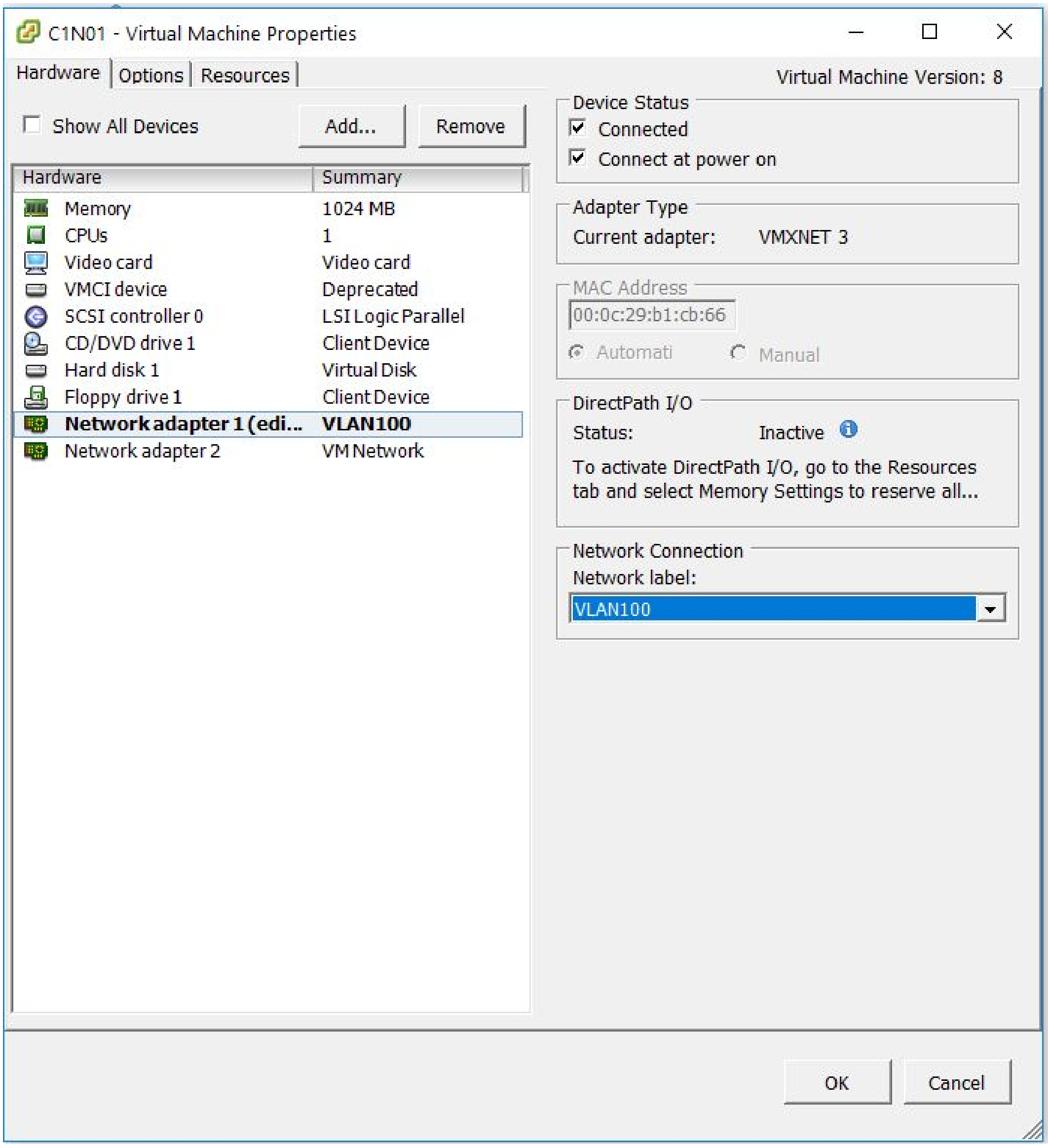

Figure C-9: Assigning Appropriate Networks to the Test VM Interfaces.

Finally, the EMANEServer needs to be assigned three interfaces:

- The VM Network (for control);
- The EMANEtrunk (for all of the TN VM traffic); and
- The EMANEdata (to communicate with other EMANEservers running on other physical machines).

This is shown in Figure C-10.

Lastly, an IP address scheme must be developed and assigned to each of these interfaces. As an example, the control interfaces for each VM could be assigned an IP such as 10.0.12.x, where x is unique for each VM. Each of the data interfaces could be assigned an IP in the 192.168.x.y range, where x represents a company within the Anglova scenario and the y represents a node within the company. The protocols/service to be tested must be configured to use the 192.168.x.y networks for data exchange.

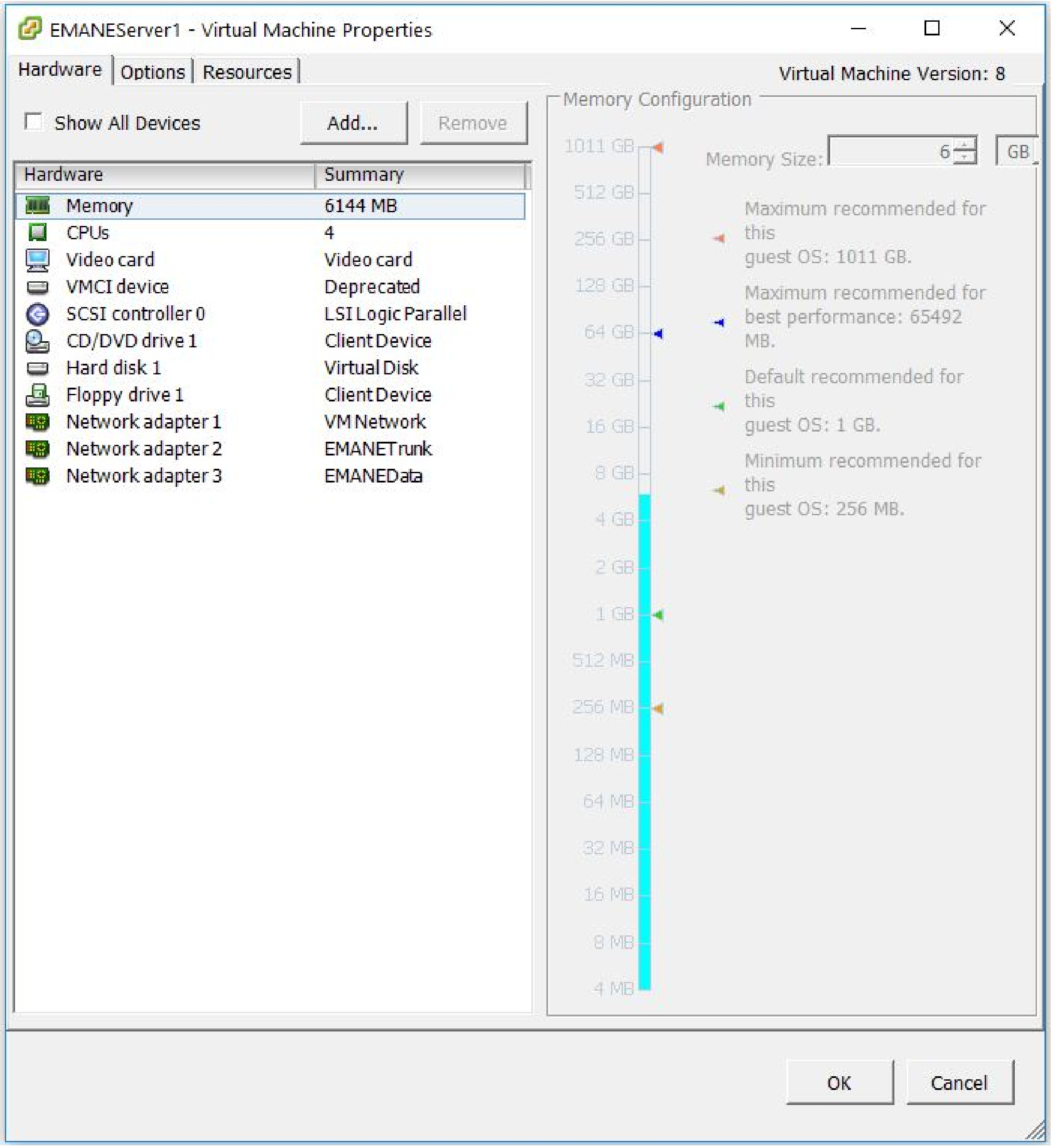

Figure C-10: Configuration of Networks for the EMANEServer VM.

C.4 VISUALIZATION TOOLS

C.4.1 Scripted Display Tools 3D (SDT-3D)

One method to visualize the Anglova scenario is by using the Scripted Display Tools (SDT-3D) and the EMANE SDT-3D client/server framework. The Scripted Display Tools (Figure C-11) are open source software developed by the Naval Research Laboratory (NRL) Protocol Engineering Advanced Networking (PROTEAN) group. SDT-3D provides real-time visualization of dynamic mobile data communication networks [6].

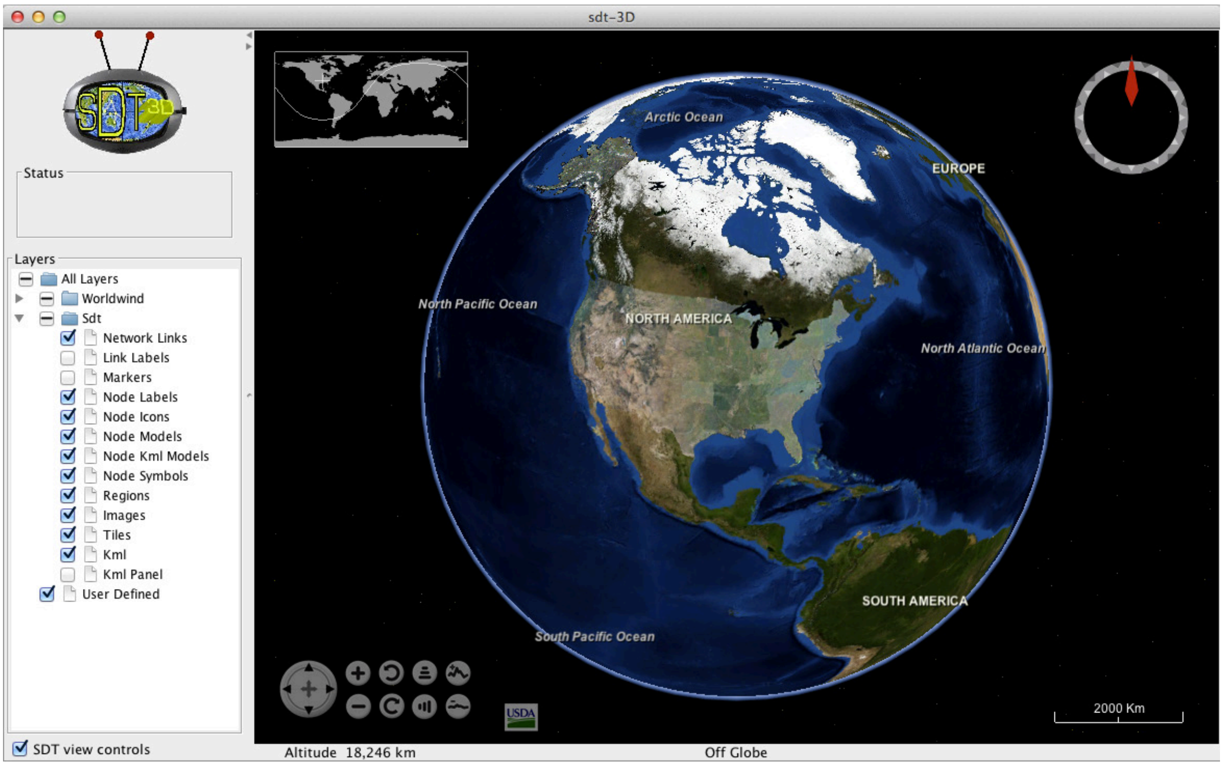

Figure C-11: Scripted Display Tools (SDT-3D).

The EMANE SDT-3D client/server framework (Figure C-12) enables the sending of visualization scripting commands to a running instance of SDT-3D to visualize a running EMANE emulation in real-time. The EMANE SDT-3D client reads location and network connectivity from the EMANE software and sends that information to the EMANE SDT-3D server. The EMANE SDT-3D server uses this information to generate and send SDT-3D scripting commands to visualize the running emulation.

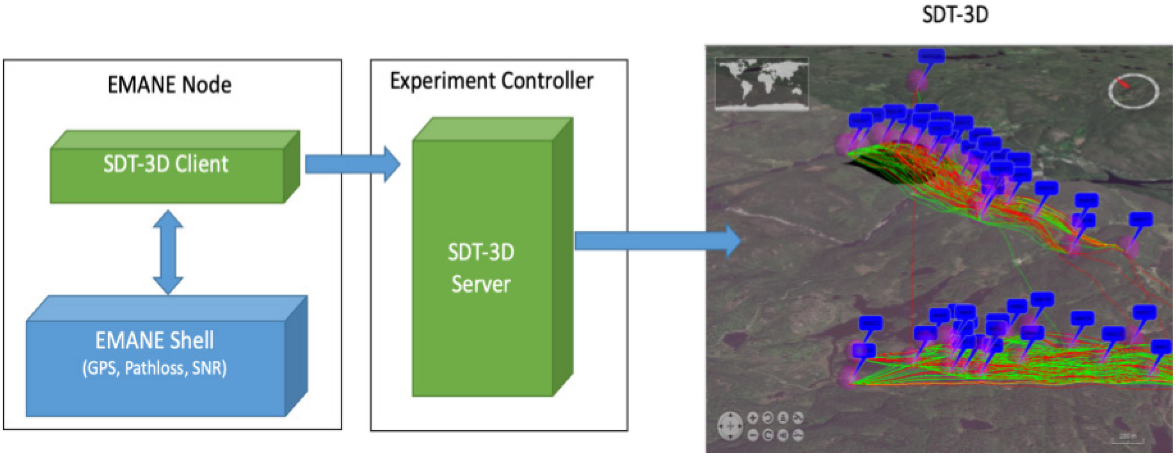

Figure C-12: EMANE SDT-3D Visualization Client/Server Framework.

Using SDT-3D and the EMANE SDT-3D client/server framework, the Anglova scenario's emulated nodes and their mobility, links and connectivity are drawn and updated dynamically on a NASA Whirlwind geographic background (Figure C-13).

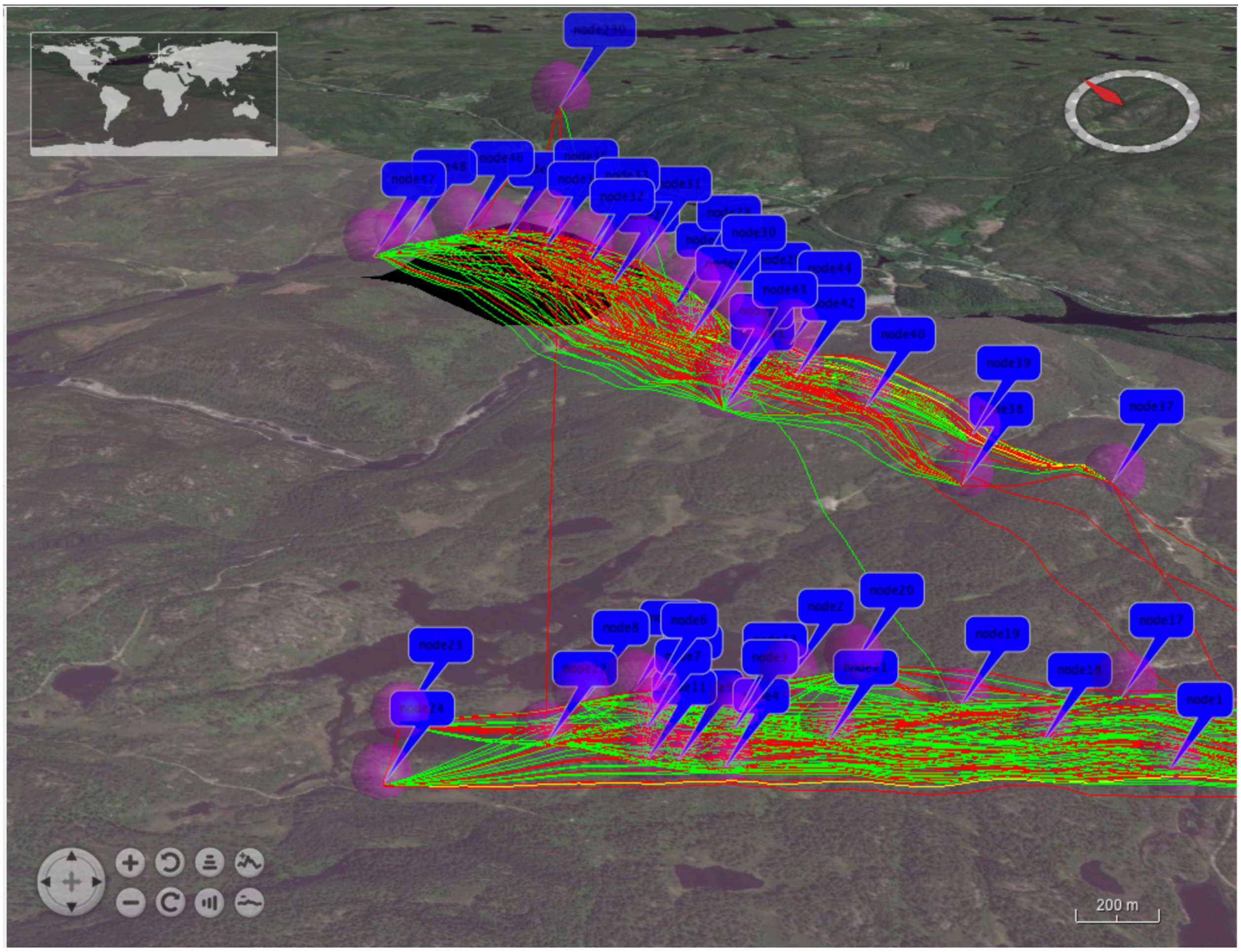

Figure C-13: Anglova Scenario Emulation Visualization in SDT-3D.

C.4.2 Mirage

Mirage is a tactical 3D visualizer based on the NASA World Wind library for Java. World Wind Java is a Software Development Kit (SDK) that adopts a set of OpenGL libraries for rendering. Mirage supports custom maps and tiles while being able to overlay NASA and United States Geological Survey (USGS) satellite imagery, topographic maps, aerial photography, Keyhole Markup Language (KML), and Collada files.

Users can interact with the 3D representation of the earth by rotating it, tilting the view, and zooming in and out. Mirage is able to display different kinds of data, from MIL-STD-2525 standard objects (Tracks, Graphical Shapes, etc.) to place names, political boundaries, latitude/longitude lines, and so forth. Mirage is able to receive real-time updates and notifications from a variety of data sources.

As part of developing the Anglova scenario, the task group also integrated the OpenStreetMap dataset and the EMANE events so that Mirage can be used to display the scenario as it is played during the course of an experiment.

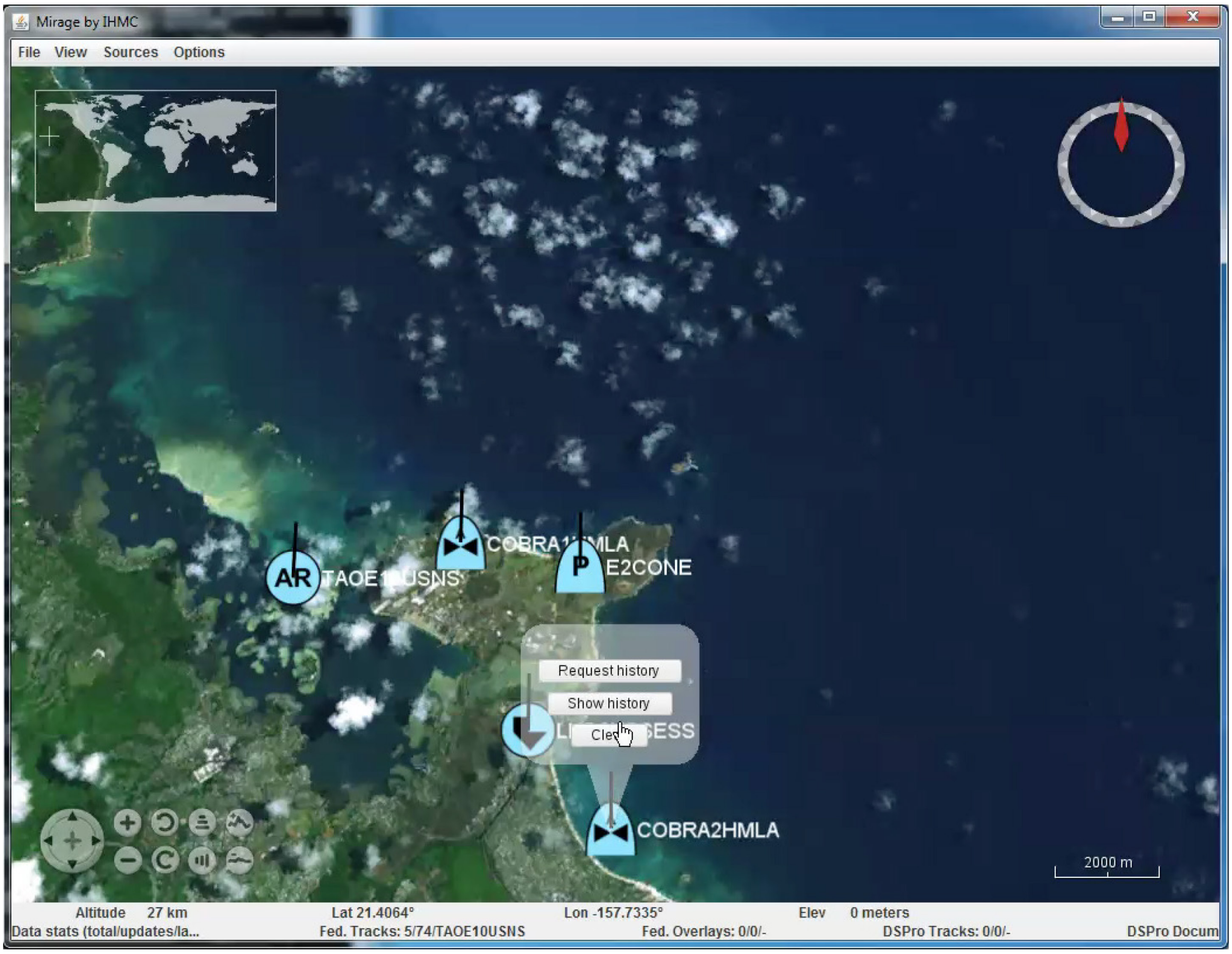

Figure C-14: Mirage Visualizer.

C.4.3 OpenStreetMap Dataset

The original location from where parts of the Anglova scenario traces were obtained was considered sensitive, resulting in the location data being anonymized and relocated to an area near Kristiansand in Norway. The coordinates were recalculated by taking Earth curvature into account so that distances were maintained within the scenario area. Actual destination location was decided based on matching topographical features. However, a drawback with relocation or movement patterns was that visualization tools could not realistically depict the movement of the vehicles over a map, as the movement would not match roadways and other elements of the terrain. As a result, the IST-124 RTG decided to generate new map data that could be used by a visualization tool while showing the movement of nodes within the Anglova scenario.

The relocated area is under web tile map tile x = 133, y = 76 on zoom level 8, coordinates (58.1298N, 7.60E) – (58.502N, 8.095E). Using the Overpass API, a slightly larger area from OpenStreetMap database was downloaded. Because of some large features this area was actually (57.74N, 6.66E) – (58.88N, 9.35E). The original area where the scenario takes place was downloaded from OpenStreetMap (OSM). Coordinates from the original area were transformed. Also all names within that dataset were transformed to Scandinavian-sounding names to hide the origin. All these modifications were simple to perform as OpenStreetMap data is XML-based and well documented, such that text-based tools could be utilized.

The two datasets were loaded into JOSM editor [7]. The section of the original terrain north of Kristiansand was removed. The terrain of the relocated scenario replaced the original terrain and then some map features, most notably the road network and waterways within scenario, were connected to surrounding areas with some artistic freedom.

After the OSM data was completed, new map tiles covering the modified area were generated. The data was imported to Postgres database with GIS extension. Map tiles were generated using litesrv renderer with OpenStreetMap Carto style as in April 2016.

The tiles are hosted at a server operated by Aalto University. Preview of the map can be seen by visiting https://www.anglova.net/. To view the modified tiles that represents the Anglova terrain within an application, the URL of https://tile.anglova.net/{z}/{x}/{y}.png can be used. The `{x}`, etc. placeholders should be replaced with application-specific ones. The format is identical to the one OpenStreetMap uses for its tile servers. All generated tiles (size 1.7 GiB) can be downloaded from https://tile.anglova.net/ist124.tiles.tar.gz for local hosting. As both the map style and map data are evolving documents there may be discontinuity at the border areas of the modified tiles.

However, many GIS and visualization applications, including the Worldwind component of SDT3D, do not support tile maps but only Open Geospatial Consortium (OGC) Web Map Service (WMS). For this reason, a WMS service is available at https://wms.anglova.net/wms supporting WMS versions 1.0.0, 1.1.1, and 1.3.0. The Anglova scenario imagery is available on the "ist124" layer. The site https://www.anglova.net contains instructions of how to configure the service into various GIS software, including SDT3D.

The modified tiles provides the correct map data to be used with tools to render the Anglova scenario environment. However, as yet, there is no areal imagery to match the modified map area. This is under study and possible options are being explored.

C.5 SCENARIO PLAYBACK TOOLS

C.5.1 EMANE Event Service Tool

In order for the Anglova scenario to progress, each node must receive the location and path loss events for each time step in the scenario. The EMANE event service is responsible for generating mobility and pathloss events for each node running within the Anglova scenario emulation. The EMANE event service takes as its input several XML configuration files as well as event files in Emulation Event Log (EEL) format and multicasts these events over a shared network channel accessible by each emulated node. Each node runs the EMANE event daemon which listens to and receives events from the event multicast channel. The event daemon's remaining role is to make events available to 'application space' by means of its 'agent' plug-ins such as the GPSd location agent. This agent is responsible for making location events available as National Marine Electronics Association (NMEA) sentences, which can serve as input to user applications (e.g., GPSd, the GPS daemon) running on the node. The EMANE event service tool's XML files are discussed below.

C.5.1.1 Event Service XML

The event service XML configuration file (Figure C-15) defines the multicast channel and interface where events will be published. It also defines the EEL generator configuration file, which contains the event parser and multicast channel configurations.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE eventservice SYSTEM "file:///usr/share/emane/dtd/eventservice.dtd">
<eventservice>
  <param name="eventservicegroup" value="224.1.2.8:45703"/>
  <param name="eventservicedevice" value="eth1"/>
  <generator definition="eelgenerator_vignette3_part1.xml"/>
</eventservice>
```

Figure C-15: Example Event Service Configuration File.

C.5.1.2 EEL Generator XML

The EEL generator configuration file (Figure C-16) defines the source EEL event file and the various parser plug-ins that will be used to parse sentences from the EEL event file.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE eventgenerator SYSTEM "file:///usr/share/emane/dtd/eventgenerator.dtd">
<eventgenerator library="eelgenerator">
    <param name="inputfile" value="vignette3_part1.eel" />
    <paramlist name="loader">
      <item value="commeffect:eelloadercommeffect:delta"/>
      <item value="location,velocity,orientation:eelloaderlocation:delta"/>
      <item value="pathloss:eelloaderpathloss:delta"/>
      <item value="antennaprofile:eelloaderantennaprofile:delta"/>
    </paramlist>
</eventgenerator>
```

Figure C-16: Example EEL Generator Configuration File.

C.5.1.3 EEL Source File

Mobility and pathloss events are stored in and parsed from EEL source files (Figure C-17). These files consist of time stamped entries for each node's GPS location (latitude, longitude, altitude) as well as time stamped entries for the pathloss values between the nodes. Mobility and pathloss events may be combined into a single file or separated into individual files.

```
0.0 nem:12350 location gps 58.170309,7.944924,2.0
0.0 nem:12350 pathloss nem:12550,0.0 nem:12650,0.0 nem:12700,0.0 nem:12750,0.0 nem:12800,0.0 nem:12850,0.0 nem:12900,0.0 nem:12950,0.0 nem:13000,0.0 nem:1305
0,0.0 nem:13100,26.2 nem:13150,26.2 nem:13250,26.2 nem:13300,26.2 nem:13350,26.2 nem:13400,26.2 nem:13450,26.2 nem:13500,26.2 nem:13550,26.2 nem:13600,26.2 n
em:13650,26.2 nem:13700,0.0 nem:13760,26.2
0.0 nem:12400 location gps 58.170309,7.944924,2.0
0.0 nem:12400 pathloss nem:12600,0.0 nem:13200,10.7 nem:13850,17.3 nem:14100,17.3 nem:14700,17.3 nem:15350,19.5 nem:15550,19.5 nem:16150,10.7
0.0 nem:12450 location gps 58.170309,7.944924,2.0
0.0 nem:12450 pathloss nem:13900,32.9 nem:15400,35.0
0.0 nem:12500 location gps 58.170309,7.944924,2.0
0.0 nem:12500 pathloss nem:13740,0.0 nem:13775,10.7 nem:14000,17.3 nem:15180,17.3 nem:15250,17.3 nem:15450,19.5 nem:16680,19.5 nem:16750,10.7 nem:16800,55.9
0.0 nem:12550 location gps 58.170309,7.944924,2.0
0.0 nem:12550 pathloss nem:12350,0.0 nem:12650,0.0 nem:12700,0.0 nem:12750,0.0 nem:12800,0.0 nem:12850,0.0 nem:12900,0.0 nem:12950,0.0 nem:13000,0.0 nem:1305
0,0.0 nem:13100,26.2 nem:13150,26.2 nem:13250,26.2 nem:13300,26.2 nem:13350,26.2 nem:13400,26.2 nem:13450,26.2 nem:13500,26.2 nem:13550,26.2 nem:13600,26.2 n
em:13650,26.2 nem:13700,0.0 nem:13760,26.2
```

Figure C-17: Example EEL File.

C.5.2 Anglova Scenario Player

As an alternative to playing back the scenario via EEL files and the EMANE Event Service Tool, we developed the Anglova scenario player, a Java-based graphical user interface, to load the scenario data files and generate the necessary EMANE events to drive the emulation. This player is showed in Figure C-18.

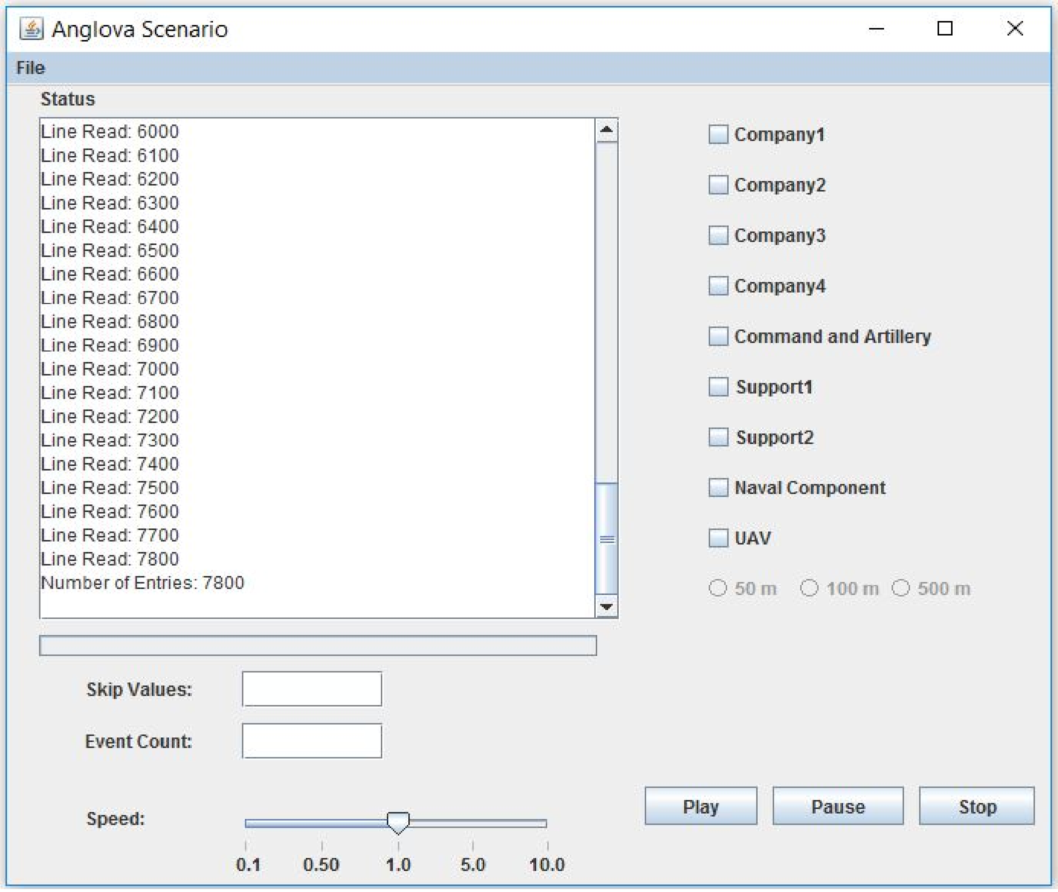

Figure C-18: Anglova Scenario Player.

The File menu item allows to load the pathloss and position files from the file system and shows the progress on the status window. Then, the events are generated by pressing the Play button. This process can be paused or stopped by using the other 2 buttons. The player also allows skipping values and counts the events that have been generated as well as regulate the speed at which the events are generated. The user can also select a subset of the companies, the naval component, and the Unmanned Aerial Vehicle (UAV), that are part of the scenario playback. For the UAV, it is also possible to select from three different altitudes. The status window provides updates about the progress of loading the event files and the playback of the scenario.

We developed some utility code in order to facilitate the generation of EMANE events to drive the emulation. EMANE receives the position and pathloss events for each node via messages that are encoded using Google ProtoBuf. This code leverages the ability to process .proto files to generate Java classes, where each class represents one event. The utility code and the underlying classes were packaged into a single .jar file so that it can provide an API for anyone wishing to generate and send EMANE events.

The following Java API describes all the steps for generating a pathloss event using the parameters target NEM id, a float array of the pathloss values, an integer array of NEM mapping, a sequence number and the uuid of the node.

The first step is generating the pathloss event builder and add the target NEM id:

```
Pathlossevent.PathlossEvent.Builder pathlossEventBuilder =
Pathlossevent.PathlossEvent.newBuilder();
```

The second step is adding the pathloss values between the target NEM and all the other NEMS, adding the pathloss values and then create a new builder:

```
Pathlossevent.PathlossEvent.Pathloss.Builder pathlossBuilder =
Pathlossevent.PathlossEvent.Pathloss.newBuilder();
pathlossBuilder.setNemId (nemMapping[i]);
pathlossBuilder.setForwardPathlossdB (pathlossValues[i]);
pathlossBuilder.setReversePathlossdB (pathlossValues[i]);
pathlossEventBuilder.addPathlosses (pathlossBuilder);
```

Create a byte array:

```
ByteString pathlossEventBA =
pathlossEventBuilder.build().toByteString();
```

Then we need to serialize the event by constructing a serial builder and setting the event id to 101 which is the identification for pathloss events that has to be manually set into a BasicEvent instance:

```
EMANEMessage.BasicEvent.Event.Data.Serialization.Builder
serialBuilder = BasicEvent.Event.Data.Serialization.newBuilder();
serialBuilder.setNemId (targetNEMId);
serialBuilder.setEventId (101);
serialBuilder.setData (pathlossEventBA);
serialBuilder.build();
```

Create a Basic event builder:

```
EMANEMessage.BasicEvent.Event.Builder evtBuilder =
EMANEMessage.BasicEvent.Event.newBuilder();
evtBuilder.setUuid (uuid);
evtBuilder.setSequenceNumber (seqNo);
BasicEvent.Event.Data.Builder dataBuilder =
BasicEvent.Event.Data.newBuilder();
dataBuilder.addSerializations (serialBuilder);
BasicEvent.Event.Data data = dataBuilder.build();
```

Convert it to a byte array, compute the length of the event and prepends the network packet with 2 bytes:

```
byte[] fullDataByteArray = evtBuilder.setData
(data).build().toByteArray();
int arrayLen = fullDataByteArray.length;
byte[] newbuf = new byte [fullDataByteArray.length + 2];
ByteBuffer bb = ByteBuffer.wrap (newbuf);
bb.putShort ((short) arrayLen);
bb.put (fullDataByteArray);
```

The last step is to create a datagram packet and send it over the EMANE event channel via UDP multicast:

```
MulticastSocket multicastSock = new MulticastSocket (45703);

DatagramPacket dp = new DatagramPacket (newbuf, newbuf.length,
_group, 45703);

multicastSock.send (dp);
```

These network packets are picked up by the EMANE Network Emulation Modules (NEMs), which then interpret the pathloss information to enforce the communications characteristics.

C.6 CUSTOMIZED PATHLOSS GENERATION TOOLS

We have also produced utility programs to generate pathloss data for some of the nodes within the Anglova scenario. Pathloss generation for the Naval Platforms in Vignette 2 and for all platforms in Vignette 3 is accomplished by using open source SPLAT! (an RF Signal Propagation, Loss, And Terrain analysis tool) [8] program using the Longley-Rice model. Location information can be generated from a motion plan template or from the locations previously entered into an EMANE Emulation (EEL) file. Antenna heights can be retrieved from the altitude information from the EEL file, or can be used as fixed antenna heights from a configuration file. Longley-Rice parameter data files are required for SPLAT! to determine RF path loss in either point-to-point or area prediction mode. Several parameter files can be specified to serve with each different network (frequency settings, relative permittivity, earth conductivity, etc.) to cope with over the sea or land communication conditions. Topographic data can be imported from various sources. Significantly better resolution and accuracy can be obtained through the use of Shuttle Radar Topography Mission-3 (SRTM-3) Version 2 digital elevation models. We have used 3-arc second SRTM data which is publicly available for over the land calculations. Ground clutter can be specified as an input parameter to pathloss generation process, we have used 30 feet average ground clutter to better simulate the effects of ground structure for over the land pathloss calculations.

While the pathloss calculations give pretty realistic results for the naval platforms and sensor network, problems occurred when calculating pathloss generation of very close units such as trucks moving in a convoy or infantry. Because the Longley-Rice (ITM) model did not give us results for objects closer than 176 meters, we have assumed 0 dB pathloss for those platforms. Also lack of exact terrain models prevented us from calculating realistic pathloss values which take into account obscuration due to buildings, plants, etc.

In particular, we developed a utility program called “ScenGenFromFile” and used it to provide recalculated path loss (as described above) as part of the Anglova scenario. The utility program is being made available with the distribution package of the IST-124 emulation environment. It can use a network plan (network_plan.csv – excel file saved as csv format) file as an input as maintained by IST-124 group to show the platforms, networks and connectivity requirements among those networks by assigning a radio (nem id is shown in green background) for the required networks as shown in Table C-1.

Table C-1: Sample Network Plan File.

id	Group	wideband1	wideband2	wideband3	wideband4	narrowband1
network	Subnets	192.168.1.0	192.168.2.0	192.168.3.0	192.168.4.0	192.168.5.0
1	company1	1	0	0	0	50
2	company1	100	0	0	0	150
3	company1	200	0	0	0	0
4	company1	250	0	0	0	0
5	company1	300	0	0	0	0
6	company1	350	0	0	0	0

id	Group	wideband1	wideband2	wideband3	wideband4	narrowband1
7	company1	400	0	0	0	0
8	company1	450	0	0	0	0
9	company1	500	0	0	0	0
10	company1	550	0	0	0	0
11	company1	600	0	0	0	0
12	company1	650	0	0	0	0
13	company1	700	0	0	0	0
14	company1	800	0	0	0	0
15	company1	850	0	0	0	0
16	company1	900	0	0	0	0
17	company1	1000	0	0	0	0
18	company1	1050	0	0	0	0
19	company1	1100	0	0	0	0
20	company1	1200	0	0	0	0
21	company1	1250	0	0	0	0
22	company1	1300	0	0	0	0
23	company1	1400	0	0	0	0
24	company1	1450	0	0	0	0
25	company2	0	1500	0	0	1550
26	company2	0	2000	0	0	2050
27	company2	0	2100	0	0	0

In this plan the leftmost "id" column corresponds to the platform depicted by that id and the green cells represents the nem id for each specific network that is deployed on the platform.

In order to generate the pathloss information the antenna heights are to be specified. The antenna heights can be fixed or variable length as some platforms may hover up and down during the scenario. The antenna height information is handled as follows. The altitude information is the mobility file location event can be used an antenna height and a platform_conf.xls file can depict default platform heights if the former information does not exist.

In this setup, the default minimum antenna heights are taken from the platform_conf.csv (excel file saved as csv format) configuration file and if a location event with higher antenna height is seen at the mobility file it is accepted as the current antenna height (see Table C-2).

Table C-2: Sample Program Configuration Initialization File.

#ID	TYPE	IMAGE	NAME	NETWORK	ANTENNA HEIGHT (meters)
1	Truck	truck	company1	0	3
2	Truck	truck	company1	0	3
3	Truck	truck	company1	0	3
4	Truck	truck	company1	0	3
5	Truck	truck	company1	0	3
6	Truck	truck	company1	0	3

#ID	TYPE	IMAGE	NAME	NETWORK	ANTENNA HEIGHT (meters)
7	Truck	truck	company1	0	3
8	Truck	truck	company1	0	3
9	Truck	truck	company1	0	3
10	Truck	truck	company1	0	3
11	Truck	truck	company1	0	3
12	Truck	truck	company1	0	3
13	Truck	truck	company1	0	3
14	Truck	truck	company1	0	3
15	Truck	truck	company1	0	3

The #ID column must match the platform ids at the network plan and the other columns are used for SDT3D visualization.

SPLAT! requires Longley-Rice model Parameter files (LRP files) for each network in order to calculate pathloss information according to the model. These files should be prepared and named as "splat_<n>.lrp" file where n is the network number in the network plan where 1 is the leftmost network column (n = 1 is wideband1, n = 2 is wideband2, … n = 14 navy uhf, etc.) in the network plan. The contents of these files are depicted in the SPLAT! Documentation [8].

A sample LRP file is shown below:

```
5.000      ; Earth Dielectric Constant (Relative permittivity)
0.001      ; Earth Conductivity (Siemens per meter)
301.000    ; Atmospheric Bending Constant (N-Units)
300.000    ; Frequency in MHz (20 MHz to 20 GHz)
6          ; Radio Climate
0          ; Polarization (0 = Horizontal, 1 = Vertical)
0.50       ; Fraction of Situations
0.50       ; Fraction of Time
0.00       ; Transmitter Effective Radiated Power in Watts or dBm (optional)
```

Please consult SPLAT! documentation for the meaning and use of this data.

Another input to the process is the topographic data .sdf files. The SPLAT! manual describes how to obtain these files, but some of the links are old or obsolete. In the current project we downloaded and used 3 arc seconds (90 m) Shuttle Radar Topography Mission (SRTM) data for the region of interest from the Web, by searching SRTM-3 via the internet. For example, webGIS internet site [9] can be used to obtain this kind of data.

The SRTM data can be changed to the format which the SPLAT! program understands by using srtm2sdf utility which comes with the SPLAT! Package [8].

Figure C-19 shows the outline of the directory structure of the provided utility program.

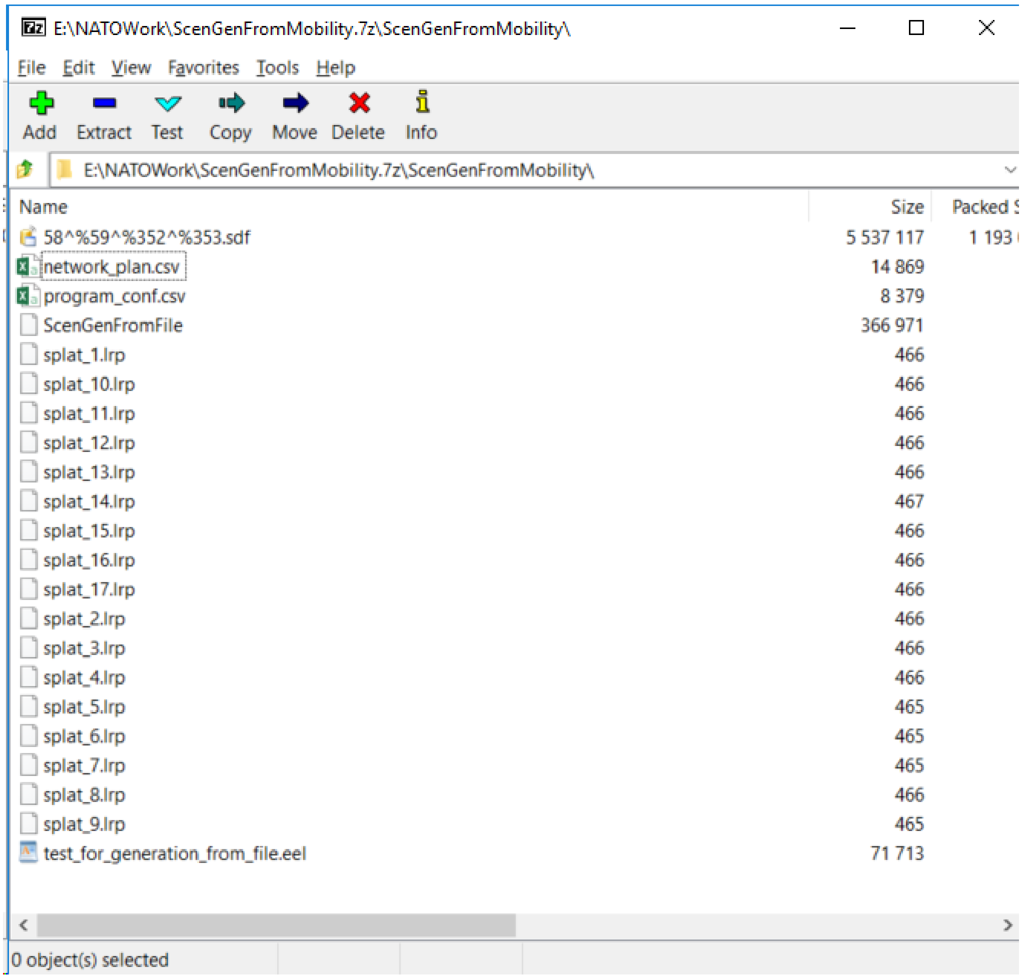

Figure C-19: ScenGenFromFile Utility Program Initial Directory Structure.

The program “ScenGenFromFile” is a Linux executable and tested to work with Centos 7 or Ubuntu 14. The file “test_for_generation_from_file.eel” is a sample input mobility file which contains mobility information of the platforms.

To use the utility program, type in the command shell:

./ScenGenFromFile <mobility_filename.eel>

The program outputs progress and when the execution finishes, the target EEL file is generated with the default name “generated_file.eel” in the same directory. The generated EEL file has the location events with altitude information as antenna height as explained before and pathloss events for all the nem ids depicted in the network plan. Sample output is shown below:

0.0 nem:17250 pathloss nem:17050,96.7 nem:17350,0.0 nem:17450,0.0 nem:17550,84.7 nem:17700,84.7 nem:17800,80.2 nem:17900,80.2 nem:18000,88.0 nem:18100,88.0 nem:18200,78.4 nem:18250,78.4 nem:18300,86.3 nem:18350,86.3 nem:18400,90.2 nem:18450,90.2 nem:18500,76.0 nem:18550,76.0 nem:18600,83.1 nem:18650,83.1 nem:18700,78.3 nem:18850,78.3 nem:20750,94.2

0.0 nem:17300 location gps 58.124413,8.035026,20.0

0.0 nem:17300 pathloss nem:17100,95.9 nem:17400,0.0 nem:17500,0.0 nem:17650,73.0 nem:17750,73.0 nem:17850,64.2 nem:17950,64.2 nem:18050,79.3 nem:18150,79.3 nem:18900,60.5 nem:20800,88.2

0.0 nem:17350 location gps 58.124413,8.035026,20.0

0.0 nem:17350 pathloss nem:17250,0.0 nem:17050,96.7 nem:17450,0.0 nem:17550,84.7 nem:17700,84.7 nem:17800,80.2 nem:17900,80.2 nem:18000,88.0 nem:18100,88.0 nem:18200,78.4 nem:18250,78.4 nem:18300,86.3 nem:18350,86.3 nem:18400,90.2 nem:18450,90.2 nem:18500,76.0 nem:18550,76.0 nem:18600,83.1 nem:18650,83.1 nem:18700,78.3 nem:18850,78.3 nem:20750,94.2

C.7 OPEN ISSUES

During the course of experimentation, four open issues were encountered with how we used EMANE in the Anglova scenario. This should be resolved with further work on the Anglova scenario. The first issue was with the RFPipe implementation of EMANE. In particular, the radio models for Anglova were first developed to use RFPipe. However, the RFPipe model, as configured and used in the Anglova scenario, does not implement any MAC algorithm and does not realize any interference effects. While the radio model allows the definition of a capacity limit as well as realize latency and reliability effects, multiple nodes are allowed to transmit at their individual rate limits and the receivers could receive, in aggregate, data rates higher than the capacity of the radio. Furthermore, RFPipe does not realize typical wireless communications effects such as hidden neighbors. RFPipe is good for point-to-point links, but cannot be used as-is for a radio network sharing a common channel.

Another issue with the radio models was discovered when using the TDMA model to emulate the narrowband links within the scenario. Unlike the RFPipe, the TDMA model includes a MAC and limits the data rates correctly. However, the TDMA implementation requires that all the NEMs using the TDMA model have synchronized clocks with an accuracy that is a function of the TDMA time slice. For the Anglova scenario, the accuracy of the Network Time Protocol (NTP) was insufficient for the TDMA model to work in a distributed deployment. The hybrid deployment works if the NEMs for all of the nodes that are part of a TDMA network are deployed within the same EMANE instance.

We observed the following issue with our reduced data rate modification of the 802.11abg model within EMANE: when multiple senders are transmitting at the maximum data rate to a single receiver, the incoming data rate at the receiver exceeds the maximum channel rate by a small fraction (as a function of the number of transmitters).

Finally, the last issue was with the EMANE event generation model, which allows a control application to generate events that control the emulation environment. These events are sent to the NEMs via UDP multicast. As mentioned earlier, the Anglova scenario has a playback component that generates the position and pathloss events to send to EMANE. Given the size of the Anglova scenario and the update rate of 1 Hz, the EMANE receiver seemed to be too slow to receive and process the events without losing some of the packets due to the UDP receive buffer being too small. This problem was solved by splitting the scenario into smaller subsets that were played back independently but still synchronized so that EMANE could support all the nodes simultaneously.

C.8 CONCLUSIONS

The Anglova scenario and all of the related files for setting up and running experiments using EMANE are available for download at http://www.ihmc.us/nomads/scenarios and also at http://anglova.net. For IST-124, having this common scenario has worked very well for efficient collaboration across multiple countries and organizations, especially when hosted in a cloud environment with tools such as DAVC to setup experiments and provide access. We hope that the details provided in this annex are useful to other researchers that are interested in using the Anglova scenario for their own experimentation.

C.9 REFERENCES

[1] AdjacentLink, LLC. Extendable Mobile Ad-hoc Network Emulator (EMANE). Internet: https://github.com/adjacentlink/emane/wiki, [May 8, 2017].

[2] Marcus, K. and Cannata, J. Dynamically Allocated Virtual Clustering Management System. Proceedings of SPIE 8742, Ground/Air Multisensor Interoperability, Integration, and Networking for Persistent ISR IV, May 2013.

[3] US Naval Research Laboratory. Multi-Generator (MGEN). Internet: http://www.nrl.navy.mil/itd/ncs/products/mgen, [May 5, 2017].

[4] Optimized Link State Routing (OLSR). Internet: http://www.olsr.org, [May 5, 2017].

[5] Marcus, K., Barz, C., Kirchhoff, J., Rogge, H., Nilsson, J., in't Velt, R., Suri, N., Hansson, A., Sterner, U., Hauge, M., Lee, K., Holtzer, A., Buchin, B Peuhkuri, M., and Mīsīrlīoğlu, L., "Evaluation of the Scalability of OLSRv2 in an Emulated Realistic Military Scenario", Proceedings of the 2017 International Conference on Military Communications and Information Systems (ICMCIS*)*, Oulu, Finland, 2017.

[6] The Scripted Display Tools www.nrl.navy.mil/itd/ncs/products/sdt, [May 8, 2017].

[7] JOSM extensible editor for OpenStreetMap (OSM). Internet: http://josm.openstreetmap.de, [May 8, 2017].

[8] Magliacane, J., SPLAT! an RF Signal Propagation, Loss, And Terrain Analysis Tool. http://www.qsl.net/kd2bd/splat.html, [May 8, 2017].

[9] WebGIS http://www.webgis.com/srtm3.html, [May 8, 2017].