Defence Research and Development Canada
Recherche et développement pour la défense Canada

Security Posture Assessment Demonstrator and Experimenter

Software Development Report

Eugen Bacic, Glen Henderson and Larry Tremblay

Defence R&D Canada – Ottawa

Canada

Contract Report
DRDC Ottawa CR 2011-012
May 2011

Security Posture Assessment Demonstrator and Experimenter

Software Development Report

Eugen Bacic
Bell Canada

Glen Henderson
Bell Canada

Larry Tremblay
Bell Canada

Prepared by:

Bell Canada
160 Elgin St., 17th Floor, Ottawa, Ontario, K2P 2C4

Project Manager: Reginald Sawilla
Contract Number: W7714-071028
Contract Scientific Authority: Reginald Sawilla

Defence R&D Canada – Ottawa

Contract Report

DRDC Ottawa CR 2011-012

May 2011

Scientific Authority

Original signed by Reginald Sawilla

Reginald Sawilla

Approved by

Original signed by for Julie Lefebvre

Julie Lefebvre
Section Head/NIO Section

Approved for release by

Original signed by for Chris McMillan

Chris McMillan
Head/Document Review Panel

Abstract

There is a recognized need within the network defence community for analysis tools that can assist in the evaluation of networks for vulnerabilities and aid in the design of robust networks. This paper provides a summary of the software development effort to create a pluggable software framework that can leverage network defence analysis tools, including advanced technical and research elements, in a unified framework in support of the design of secure and robust networks. The technology components for this framework, specifically RapidMiner and a series of solution specific software elements, are described in the context of the operational environment configuration in which these analysis tools are deployed. A high-order conceptual workflow is provided that details the processing logic, dependencies and data flow that constitute the operational practices of the network analysis framework. Build instructions for the creation and maintenance of the tool set are provided and a tutorial oriented toward the target user community describes how this analytical tool can be used to perform an assessment of a sample network and how to interpret the results using the integrated visualization capabilities. A series of suggestions for future enhancement of the tool set are similarly provided.

Résumé

Il est reconnu que la collectivité de la défense de réseaux a besoin d'outils d'analyse pouvant aider à évaluer les vulnérabilités des réseaux et à concevoir des réseaux solides. Cette étude résume les travaux de développement de logiciels visant à créer un cadre logiciel enfichable pouvant exploiter les outils d'analyse de la défense de réseaux, y compris des éléments techniques et de recherche avancés, dans un cadre unifié pouvant soutenir la conception de réseaux sûrs et solides. Les composants technologiques de ce cadre, en particulier RapidMiner et une série d'éléments logiciels axés sur des solutions, sont décrits en fonction de la configuration du milieu opérationnel dans lequel ces outils d'analyse sont utilisés. Un flux conceptuel des travaux de poids fort présente la logique du traitement, les dépendances et le flux des données qui constituent les pratiques opérationnelles du cadre d'analyse des réseaux. Des instructions relatives à la création et à l'entretien de la trousse à outils sont énoncées, et un tutoriel axé sur les utilisateurs visés explique comment cet outil d'analyse peut servir à évaluer un modèle de réseau et comment interpréter les résultats à l'aide des capacités intégrées de visualisation. Des améliorations possibles à la trousse à outils sont également proposées.

This page intentionally left blank.

Table of contents

This page intentionally left blank.

1 Introduction

This software development report provides a high-level description of the Security Posture Assessment Demonstrator and Experimenter (SPADE), a framework for a software-based research environment for the modelling of network architectures and associated security vulnerabilities. This document details the primary technology components, the high-level module design and the necessary build procedures to instantiate the framework from its source elements. This document also provides a tutorial for the operation of the framework in a simulated network environment.

The end goal of this software development effort is the creation of a set of modelling and analysis tools with the capability to support defence research regarding network security posture. This work builds upon previous work that:

- identified and prioritized a set of requirements for this analysis tool set; and
- developed a series of network models which will be used to demonstrate and test the analysis tool set.

This report should be read in the context of the supplemental project documentation authored during the course of this effort, including:

- The *System Description and Work Plan* (note that this document also enumerated a priorities set of solution requirements);
- *Network Scenarios Architecture for Security Research Testing*; and
- *Solution Development Document*, an interim document detailing architectural design decisions and approaches (January 2010).

These documents are meant to serve as a design and development compendium regarding the development work that has been done to date on the the analytical tool set, support the maintenance of the existing framework and guide the development of future enhancements and capabilities.

2 Technology Overview

This section describes the environments that are relevant to the SPADE toolset. Specifically, these environments include:

- The **Operational Environment**: The environment in which the tools use used by security professionals and research experts.
- The **Development Environment**: The environment in which the SPADE code elements are built into their operational equivalents and are packaged into the environment; and

It is significant to note that both the Development and Operational environments coexist on the same platform. That is, there is simply an area of the SPADE deployment with all of the source code for the compiled (Java) elements of SPADE, while all of the Python elements are directly available for change in place.

2.1 Component Overview

This section provides a description of the significant solution components that have been leveraged to create the SPADE framework. The collection of components includes external software dependancies, system tools and languages, defined standards and existing research elements. Where appropriate, these components are discussed in the content of the environment or usage to which the component applies, specifically, as a component necessary for the development environment or for the operational environment.

2.1.1 RapidMiner

RapidMiner (http://rapid-i.com) is a framework for performing machine learning, data mining, and data modelling. An open-source project, RapidMiner is designed to be extensible to encompass virtually any data modelling task, whether via modifications to the RapidMiner application itself or via plugins (called operators in RapidMiner nomenclature.) For the SPADE environment, two operators were created for RapidMiner: one for invoking the MulVAL tool, and another for invoking the AssetRank tool. These operators allow users to create RapidMiner projects to simply invoke MulVAL and AssetRank, or to also use other built-in operators available in order to perform pre- or post-processing on data being analysed by those tools.

2.1.2 VMWare

The SPADE environment consists of several basic tools, which have associated with them a large set of dependencies. This entails several possible compatibility issues with those

elements that need to be addressed for any potential user. Also, the amount of time needed to set up the SPADE environment to a useable state is not insignificant. Given these, it was decided that creating a complete operating environment within a virtual machine would be the simplest course of action, providing a simple means for any user to acquire and deploy the SPADE environment. This virtual machine is distributed as a VMWare image, which can then be used by anyone by simply installing VMWare Fusion , a free product (or using VMWare Workstation if the user already holds a licence for that product.)

2.2 Operational Environment

The SPADE operational environment consists of the NetworkModeler application and an installation of RapidMiner, along with accessible versions of the tools written in the Python language.

2.2.1 Tools

Operating System

The guest operating system in the virtual machine is 64-bit Ubuntu 8.04.4 (Hardy Heron).

RapidMiner

The virtual machine contains an installation of RapidMiner version 4.6. In a future release, RapidMiner version 5.0 will be included, but some changes to the RapidMiner plugins are required before this upgrade can occur.

NetworkModeler

The NetworkModeler tool is used to create network models in a graphical environment, giving the user the capability to create sites which contain zones (analogous to subnets). A zone contains machines, and a machine can be configured with software and services. Access control lists can be defined to indicate connectivities between sites, zones, and machines. The saved XML output of NetworkModeler is used by other elements in the SPADE environment (see below) to produce other output forms that are used by other tools within the environment (e.g. NetworkModeler XML to MulVAL predicates for MulVAL attack graph processing.)

Python Utilities

There are several utilities written in the Python language that are deployed to support the SPADE environment:

- Scripts to generate MulVAL predicates from NetworkModeler XML, which also pulls in vulnerabilities from the NVD to be written to the MulVAL file

- Scripts to populate a MSSQL database with the contents of a NetworkModeler XML file, and to read the database back into XML format
- A script to partially convert a NetworkModeler XML file into Prolog predicates in order to determine connectivities between entities in a network model (haclWriter)

MulVAL

The most current version of MulVAL (at the time the virtual machine was packaged) is included in the SPADE environment

AssetRank

The most current version of AssetRank (at the time the virtual machine was packaged) is included in the SPADE environment.

XSB

The most current version of XSB (version 3.2, at the time the virtual machine was packaged) is included in the SPADE environment.

National Vulnerability Database

A local copy of the XML files for the National Vulnerability Database (NVD) is included in the SPADE environment.

Common Platform Enumeration

A local copy of the XML file for the Common Platform Enumeration (CPE) dictionary is included in the spade environment.

2.2.2 Configuration

Deployment of SPADE tools is split between two areas:

1. */usr/local/SPADE*
2. */home/user/SPADE*

Those tools that were not written by Bell are deployed in the in */usr/local/SPADE*. These include AssetRank, MulVAL, RapidMiner, and XSB. Those tools that were written by Bell are deployed in */home/user/SPADE*. These include NetworkModeler and the Python utilities as described previously. In addition to those tools, previously defined sample network models are deployed to this location as well. Due to the potential of frequent updates, the local copies of the NVD and CPE dictionary are stored in */home/user/SPADE*.

The */etc/profile* file contains additions to the system environment variables required to support the SPADE environment. These generally include additions to the PATH variable, and declarations of environment variables relevant to the various tools deployed within the SPADE environment.

2.2.3 Accounts

Under Ubuntu, a single account, ”user”, is the only active account for the operating system.

2.3 Development Environment

The components listed in this section are necessary for the creation of the SPADE framework from its constituent code base.

2.3.1 Java

Within the SPADE environment, the NetworkModeler tool and the RapidMiner operators are written in the Java language. As such, they require a development environment to be present in order to allow changes to be made to these elements. Included in the SPADE virtual machine is the complete development environment necessary. The development environment consists of:

- NetBeans v6.9, an integrated development environment (IDE) for Java
- Ant 1.8, a build tool used by NetBeans to build the source code into functional executable units
- The Java source code for NetworkModeler and the RapidMiner plugins
- NetBeans projects for NetworkModeler and the RapidMiner plugins
- All external dependencies (libraries) required by NetworkModeler and the RapidMiner plugins (see Tools, below, for details)

When the NetworkModeler tool is built from within the NetBeans environment, the newly-built distribution is copied to the NetworkModeler live directory (/home/user/SPADE/NetworkModeler) as part of the build process. The RapidMiner operators, on the other hand, must be manually copied by the user after being built (*sudo cp /home/user/SPADE/src/RapidMiner/dist /usr/local/SPADE/rapidminer/lib/plugins*). This is due to the location of RapidMiner being in the */usr* directory tree, which requires root privilege to copy into.

2.3.2 Tools

These tools and libraries are the dependencies upon which the Java tools in the SPADE environment rely.

RapidMiner

- JUNG (Java Universal Network/Graph Framework) for graph drawing and manipulation
- Apache Commons libraries (required by JUNG)
- RapidMiner common libraries to allow creation of RapidMiner plugins

NetworkModeler

- JUNG (Java Universal Network/Graph Framework) for graph drawing and manipulation
- Apache Commons libraries (required by JUNG)
- JDOM (Java Document Object Model) for XML document creation and handling
- OpenCSV for CSV file handling
- Log4j for logging

Other

- Ant 1.8 to manage the build process

2.3.3 Python

Several of the elements within the SPADE environment are written in the Python language as scripts for accomplishing various tasks. As an interpreted language, no specialized development environment is necessary for making changes to these scripts. Simply edit the source code in a text editor as desired.

2.3.4 Tools

These tools and libraries are the dependencies upon which the Python tools in the SPADE environment rely

haclWriter

- XSB

MSSQL utilities

- pymssql for communicating with MSSQL databases from Python
- FreeTDS (a dependency of pymssql)

MulVAL generation

- XSB

Note that MulVAL as an independent research project may define additional software dependancies as that project continues to refine and mature its logic programming based approach to vulnerability analysis.

2.3.5 Configuration

The SPADE development environment is deployed in the same directory hierarchy as the operational environment. Specifically, the code is contained in the */home/user/SPADE/src* directory. Under this directory hierarchy are two subdirectories:

- */home/user/SPADE/src/NetworkModeler* - the NetworkModeler application source code
- */home/user/SPADE/src/RapidMiner* - the RapidMiner operator source code

Each of these directories contains a NetBeans project for the NetBeans IDE. The NetBeans IDE itself has been installed to a convenient location (in this case, */home/user/netbeans-6.9*).

All Python code is checked out directly to */home/user/SPADE/mssql*, */home/user/SPADE/-mulvalgen*, and */home/user/SPADE/haclWriter*.

2.3.6 Accounts

Under Ubuntu, a single account, ”user”, is the only active account for the operating system.

3 Module Desccription

This section describes the high-level SPADE components and how they are combined to form a uniform experimentation environment suitable for network modelling and security experimentation.

3.1 Conceptual Workflow Overview

The process by which a user begins from a network concept to the output from MulVAL/AssetRank is described below:

1. User begins with the NetworkModeler application and defines sites and zones (subnets) for the network.
2. Machines are then added to zones.
3. Machines are populated with software, services, and accounts.
4. ACL entries are added to sites/zones/machines to indicate connectivities.
5. The NetworkModeler connectivity walk feature can be used to ensure that HACLs are correctly implemented.
6. MulVAL predicates are generated.
7. RapidMiner is called to invoke MulVAL to generate the attack graph.
8. If desired, AssetRank may also be invoked to generate rankings on the attack graph.
9. The generated attack graph is loaded into NetworkModeler for analysis.
10. Iterate as necessary.

3.2 High Level Component Overview

There are several components within the SPADE environment that work in sequence to provide the user with an end-to-end process from model to attack graph. These components are briefly described below:

- NetworkModeler is a standalone application that allows users to graphically generate networks using a simplified site/zone/machine visualisation notation that is consistent with the MulVAL modeling paradigm.

- haclWriter is a component that takes the current NetworkModeler model, and does a partial conversion of it to Prolog predicates, which are then analysed to allow the user to ”walk” the connectivities between elements in the model to validate correctness
- generateMulval is a component that takes the current NetworkModeler model, and does a complete conversion of it to MulVAL predicates. This includes analysing the software in the network against the NVD, and including all vulnerabilities that are found in the MulVAL output
- RapidMiner is a standalone application that is used to invoke the MulVAL and optionally AssetRank packages to perform analysis on the generated predicates. Users may use other features of RapidMiner to manipulate the predicates/data as desired before or after invoking MulVAL/AssetRank

In addition to those components, there is the MSSQL component that allows the user to populate a MSSQL database with the contents of a NetworkModeler XML model, and write the contents of such a database back to XML format. This allows the user to perform Transact-SQL operations in batch to quickly manipulate elements of a model (e.g. change all instances of Firefox v2.6 to v3.0).

3.3 Component Details

The following sections describe in more detail the components within the SPADE environment. Each component is given in terms of:

1. What its role is.
2. What inputs it takes.
3. What outputs it creates.
4. What are the dependencies on the component.
5. What other components it communicates with during its operation.

3.3.1 NetworkModeler

What it Does

- Allows the user to create and manipulate network models
- Define connections between sites and zones and machines
- Define what software, services, users are on machines

- Define ACLs between sites, zones, machines
- Gives the user non-commandline access to other tools in the system (generating MulVAL, RapidMiner)
- Visualises attack graphs that have been generated for the network

Inputs

- Using the GUI to create models
- Open XML models created by whatever means outside of NetworkModeler
- Open attack graphs generated by MulVAL or AssetRank

Outputs

- Models to persistent storage in XML format
- Models converted into MulVAL predicates to persistent storage
- Visualisation of connectivity paths within the network (connectivity walk)
- Visualisation of attack graph paths through the network (after analysis)

Dependencies

- haclWriter.py for walking connectivity
- generateMulval.py for creating MulVAL predicate files
- MulVAL and/or AssetRank for generating attack graphs to load

Intercomponent Communication

- Directly invokes haclWriter and reads its output for connectivity walking
- Directly invokes generateMulval, but does not do anything directly with its output

3.3.2 haclWriter

What it Does

- Takes a NetworkModeler XML model and partially converts it to MulVAL predicates
- Invokes XSB on those predicates to build a list of connectivity that is possible to/from a given machine
- Outputs that connectivity list for the invoker to read

Inputs

- A NetworkModeler XML model

Outputs

- A list of connectivity paths within the network

Dependencies

- Python, including Python XML libraries
- NetworkModeler to generate the input model
- XSB to analyse the generated predicates

Intercomponent Communication

- Directly invokes XSB
- Output is written to stdout, which is captured by NetworkModeler as its input

3.3.3 generateMulval

What it Does

- Takes a NetworkModeler XML model and converts it to a proper MulVAL predicate file
- Compares the hardware and software entities in the XML file against the NVD, and includes all vulnerabilities found in the MulVAL file

Inputs

- A NetworkModeler XML model

Outputs

- A MulVAL predicate file, written to persistent storage

Dependencies

- Python, including Python XML libraries
- NetworkModeler to generate the input model
- NVD XML files to read to do vulnerability matching

Intercomponent Communication

- None

3.3.4 MulVAL Invoker

What it Does

- This is an operator for the RapidMiner framework that takes a given MulVAL predicate file and invokes MulVAL against it

Inputs

- A MulVAL predicate file which is read from persistent storage

Outputs

- An attack graph, as a pair of CSV files containing arcs and vertices describing the attack graph, written to persistent storage
- The attack graph arc and vertex data is also available in RapidMiner data structures which can be passed to other RapidMiner operators

Dependencies

- RapidMiner
- MulVAL

Intercomponent Communication

- The RapidMiner data structures containing the attack graph are made available as inputs to other RapidMiner operators

3.3.5 AssetRank Invoker

What it Does

- This is an operator for the RapidMiner framework that takes a given AssetRank predicate file and invokes AssetRank against it

Inputs

- An attack graph represented as arcs and vertices, contained in RapidMiner data structures (i.e. the output from the MulVAL operator)

Outputs

- An attack graph, as a pair of CSV files containing arcs and vertices describing the attack graph, written to persistent storage
- The attack graph arc and vertex data is also available in RapidMiner data structures which can be passed to other RapidMiner operators

Dependencies

- RapidMiner
- AssetRank

Intercomponent Communication

- The data structures containing the attack graph arcs and vertices are passed in to the AssetRank operator from the MulVAL operator

3.4 Microsoft SQL Import and Export

In addition to the workflow described above, Python scripts for populating the content of a NetworkModeler XML file into a Microsoft SQL database, and transferring the content of the database back into XML format are provided. These scripts provide an additional layer of utility whereby Transact-SQL commands may be used in batches to make changes quickly if the user is familiar with SQL. The process by which a user goes from a NetworkModeler XML file into a MS SQL database and back again includes:

1. Create a database on the MS SQL server to be used
2. Use the *mkdatabase.sql* script to create the required table structure
3. Populate the database from a NetworkModeler XML file
4. Perform changes to the contents of the tables as desired
5. Rebuild the NetworkModeler XML file from the database
6. Use the changed XML file as desired

3.4.1 Preparing the Database

The *mkdatabase.sql* script is found in */home/user/SPADE/mssql*, along with all of the other SQL scripts. On the target MS SQL server, create a database. Copy the *mkdatabase.sql* file to the target MS SQL server and execute it to create the required tables in the database. Be sure to attach a user and grant read and write access to the database.

3.4.2 XML to SQL

Once the database and its tables have been created, a NetworkModeler XML script may be imported into it. The *xml2sql.py* script will perform the translation from XML markup and insert the data into the appropriate tables. The *xml2sql.py* script takes several parameters:

```
usage:
xml2sql.py
--db database name
--user database user
--pass database password
--addr database address
[--port database port (default 1433)]
xml input_file
```

An example of the invocation would be:

python xml2sql.py –db mydatabase –user dbuser –pass 12345 –addr mssql.example.com myNetwork.xml

In this example, the address of the MS SQL server is *mssql.example.com* (this can also be an IP address). The database that the data will be populated into is *mydatabase*, which should already have been prepared as given in section 3.4.1. The user to authenticate to the database is *user* using password *12345* (this user must have read and write access granted to the database). The *port* is optional, and need only be used if the MS SQL server is listening on a port other than 1433.

3.4.3 SQL to XML

Once the database has been manipulated as desired, the NetworkModeler XML file can be rebuilt from the table data. The *sql2xml.py* script will perform the translation from tables back into XML markup. The *sql2xml.py* script takes several parameters:

```
usage:
sql2xml.py
--db database name
--user database user
--pass database password
--addr database address
[--port database port (default 1433)]
xml output_file
```

The parameters for *sql2xml.py* are as given in section 3.4.2.

4 Build Process

This section describes the process used to build the various components within the SPADE environment. Note that this does not cover elements such as RapidMiner, XSB, MulVAL, or AssetRank. Refer to the documentation included with those packages for their specific build instructions.

4.1 Creating SPADE Components

There are two distinct sets of components for deployment within the SPADE environment:

1. components written in Java; and
2. components written in Python.

The following sections will describe the process for creating and deploying these components.

4.1.1 Java Components

The two components written in Java are the NetworkModeler application, and the MulVAL and AssetRank operators for RapidMiner. Each follows a similar basic process for building, with slight differences in deployment. The first step is to acquire the source code, which is by default included in the */home/user/SPADE/src* directory. Once the code is present, the NetBeans Integrated Development Environment (IDE) can be found in the Applications->Programming menu of the Ubuntu operating system. Launch NetBeans. Once running, NetBeans may already have the NetworkModeler and RapidMiner projects loaded and ready. If this is not the case, load the project to be modified via the File->Open Project menu. By default, the projects are in */home/user/SPADE/src/NetworkModeler* and */home/user/SPADE/src/RapidMiner*. Once loaded, each project can be built by right clicking on its name in the Projects tab on the left side of the IDE, and selecting ”Build” from the menu. See 4.2.1 for deployment details.

4.1.2 Python Components

The components written in Python (haclWriter, generateMulval, and MSSQL utilities) are entities that do not require compiling as the Java components do. Instead, the individual files may be edited directly in their directories (*/home/user/SPADE/haclWriter*, */home-/user/SPADE/mulvalgen*, and */home/user/SPADE/mssql*). Any changes made will be immediately reflected in the next invocation of the modified script.

4.2 Deploying SPADE Components

This section details the deployment of the various programmed elements of the SPADE environment.

4.2.1 Java Elements

When the NetworkModeler project is built, Ant build script for that project includes a step that copies the contents of */home/user/SPADE/src/NetworkModeler/dist* directory to */home/user/SPADE/NetworkModeler*. That is, whenever the NetworkModeler project is built from within NetBeans, it is deployed automatically.

Due to RapidMiner being deployed in the /usr hierarchy, root privilege is required to copy the files that are created during the build process. As such, the Ant build script for the RapidMiner operators cannot automatically copy after build. The user must copy the .jar files manually using the *sudo* prefix to the command (*sudo cp /home/user/SPADE/src/RapidMiner/dist /usr/local/SPADE/rapidminer/lib/plugins*). For convenience, the **copyRM** script has been created in */home/user/SPADE*, which contains the correct command to copy the RapidMiner files. Executing the script will cause the user to be prompted for their password, and the files will be copied after the password is supplied.

4.2.2 Python Elements

The MSSQL, MulVAL generation, and haclWriter elements should already be checked out to their appropriate directories under */home/user/SPADE/*. No specific deployment steps are required for these components, as any changes to them are made in-place.

4.3 Code Overview

4.3.1 RapidMiner

This section details the Java packages present in the RapidMiner project. It is intended to guide developers in finding areas of code for changes, not as a complete roadmap of the project's code.

package ca.gc.drdc_rddc.rapidminer.common

- Contains code elements that are common to the AssetRank Invoker and MulVAL invoker

package ca.gc.drdc_rddc.rapidminer.assetrank

- Contains the AssetRank Invoker code

package ca.gc.drdc_rddc.rapidminer.mulval

- Contains the MulVAL Invoker code

package ca.gc.drdc_rddc.rapidminer.visualization

- More or less defunct now, was for visualizing attack graphs after generation

4.3.2 NetworkModeler

This section details the Java packages present in the NetworkModeler project. It is intended to guide developers in finding areas of code for changes, not as a complete roadmap of the project's code.

package ca.gc.drdc_rddc.mma

- Contains the main application class

package ca.gc.drdc_rddc.mma.actions

- Classes containing actions (that should be moved elsewhere)

package ca.gc.drdc_rddc.mma.actions.menu

- Classes representing the actions associated with menu bar items

package ca.gc.drdc_rddc.mma.attackgraph

- Classes for reading and handling attack graphs

package ca.gc.drdc_rddc.mma.attackgraph.data

- Classes for data elements of the attack graph

package ca.gc.drdc_rddc.mma.cpe

- Classes for handling the CPE dictionary

package ca.gc.drdc_rddc.mma.document

- Classes describing the NetworkModeler document in XML form and in-memory form

package ca.gc.drdc_rddc.mma.frame

- Classes for the main application frame and its subdivisions

package ca.gc.drdc_rddc.mma.graph

- Classes for the network graph display
- Support classes

package ca.gc.drdc_rddc.mma.graphics

- Icon list and handling (for display on the graph)

package ca.gc.drdc_rddc.mma.listeners

- Listener classes for GUI elements

package ca.gc.drdc_rddc.mma.logging

- Classes supporting logging

package ca.gc.drdc_rddc.mma.menu

- Classes representing the application menu bar

package ca.gc.drdc_rddc.mma.model

- The ModelElement (derived from NetworkElement) class
- Support classes

package ca.gc.drdc_rddc.mma.account

- The AccountElement (derived from NetworkElement) class
- Actions for creating/editing accounts
- Dialog to support creating/editing accounts
- Support classes

package ca.gc.drdc_rddc.mma.acl

- The AclElement (derived from NetworkElement) class
- Actions for creating/editing ACLs
- Dialog to support creating/editing ACLs
- Support classes

package ca.gc.drdc_rddc.mma.machine

- The MachineElement (derived from NetworkElement) class
- Actions for creating/editing machines
- Dialog to support creating/editing machines
- Support classes

package ca.gc.drdc_rddc.mma.service

- The ServiceElement (derived from NetworkElement) class
- Actions for creating/editing services
- Dialog to support creating/editing services
- Support classes

package ca.gc.drdc_rddc.mma.share

- The ShareElement (derived from NetworkElement) class
- Actions for creating/editing shares
- Dialog to support creating/editing shares
- Support classes

package ca.gc.drdc_rddc.mma.site

- The SiteElement (derived from NetworkElement) class
- Actions for creating/editing sites
- Dialog to support creating/editing sites
- Support classes

package ca.gc.drdc_rddc.mma.software

- The SoftwareElement (derived from NetworkElement) class
- Actions for creating/editing software
- Dialog to support creating/editing software
- Support classes

package ca.gc.drdc_rddc.mma.zone

- The ZoneElement (derived from NetworkElement) class
- Actions for creating/editing zones
- Dialog to support creating/editing zones
- Support classes

package ca.gc.drdc_rddc.mma.networkelement

- The base NetworkElement class from which all other NetworkElements are derived

package ca.gc.drdc_rddc.mma.templates

- Contains classes for handling templates

package ca.gc.drdc_rddc.mma.util

- Contains utility classes for various things

package ca.gc.drdc_rddc.mma.xsb

- Contains classes used for haclWriter and connectivity walk

5 Tutorial

In this section, two tutorials will be presented:

1. Creating a simple model using NetworkModeler
2. Using NetworkModeler to create and load an attack graph

5.1 Creating a Model in NetworkModeler

In this tutorial, a simple network with three machines will be created. Begin by launching NetworkModeler from the desktop. Once NetworkModeler has loaded, select *File* $\rightarrow$ *New* on the main menu. There is now a blank network, ready to be populated.

Note: There is a copy of this tutorial 3host network, *3host.xml*, in */home/user/SPADE/networks/3host.tutorial.master* to examine and compare against.

5.1.1 Fully Qualified Names

Within NetworkModeler, a site, zone, or machine is generally referred to by its fully qualified name (FQN) when creating elements that refer to the particular element. In this case, a FQN follows the format *site_zone_machine*. For example, the workstation machine that will be created first in this tutorial has a fully qualified name of *3host_network_workstation*. Note that when creating an element the name given is not the FQN, just the simple name of the element (*workstation* in this example). Fully qualified names are used when creating other elements such as file shares, ACLs, etc. that refer to elements that must be uniquely identified.

Note: Names of all types are case-sensitive. The names *3host_Internet_workstation* and *3host_internet_workstation* are not considered the same by the system. Mismatched case will lead to features such as Connectivity Walk not working correctly, attack graph analysis not finding any attack paths (or not finding all possible attack paths), etc. It is very important to be consistent.

5.1.2 Creating a Site and Zone

The first step in creating the network is to create a site to represent the location of the machines.

- Right click on the graph pane (or *Untitled* node in the tree view) and select *New Site*
- Give the site a name (e.g. *3host*) in the dialog that appears

Now, a zone to represent the subnet that the machines are present in must be created.

- Right click on the graph pane (or the *network* node in the tree view) and select *New Zone*
- Fill in a zone name (e.g. *network*), and optionally a network IP and netmask

At this point, the zone and site will be redrawn on the graph. This is a good time to save your model.

- Select *File* → *Save*
- Create a directory for the model
- For convenience, the directory /home/user/SPADE/networks/3host exists and is empty except for the file specifying the location of the attacker in the 3host network (see 5.2)
- Give the model a name, e.g. 3host.

5.1.3 Creating and Populating Machines in the Zone

In this section, three machines will be created in the network zone: a workstation, a file server, and a web server. First, create the workstation:

- Right click within the blue zone area (or the *network* node in the tree view) to bring up its context menu
- Select *New Machine*
- Fill in a machine name (e.g. *workstation*). As above, only give *workstation* as the name here. The FQN for the element is derived from the fact that *workstation* is nested within *zone* which is in turn nested within *3host*
- Use the Type dropdown to select type Workstation
- You may either type in a CPE for the operating system directly, or use the ... button to invoke the CPE builder (see 5.1.5.1)
- Whichever method used, the CPE for workStation's operating system is *cpe:/o:microsoft:windows_xp:-:sp2:* (Microsoft Windows XP Service Pack 2)
- You may optionally add a CPE for the hardware platform of a machine. This is most commonly useful for appliances (routers)
- You may set a count of how many machines like this are present in the zone (useful if an organisation has a standard image for workstations, for example)

Now, create a user account on the workstation.

- Select the workstation and right click on it (or right click on the *workstation* node in the tree view) to bring up its context menu
- Select *New Account*
- Fill in the account name (e.g. Bob) and use account type *user*, then select OK

The workstation is now complete. Next, create a webserver in the zone. Repeat the steps used to create the workstation machine, with the following changes:

- The machine must have a different name (e.g. webServer)
- The machine type is Server
- the operating system type can be any server OS (e.g. Microsoft Windows Server 2003, any Linux distribution, etc.)

The webserver now needs a user account. Create an account on the webserver using the same method as for the workstation, except that the account should be of type *root*. Once the root account is defined, the web server service can be addded:

- Select the webServer machine and then right click on it (or right click on the *web-Server* node in the tree view) to bring up its context menu
- Select *New Service*
- Give the service a name (e.g. Apache)
- Use the *Run As* dropdown to select the user account that the service runs as (i.e. the account created in the previous step)
- Enter the CPE for the Apache HTTP Server version 2.2.2 (*cpe:/a:apache:http_server:2.2.2::*)
- Enter the port number that the service runs on (e.g. 80)
- Enter the name of the protocol used by the service (i.e. http)
- The Type dropdown should be left blank for this service

Note: *If a server listens on the same service on several ports, a service entry must be created for each port that is being listened on.*

The webServer machine is now complete. Finally, create a file server in the zone. Repeat the steps used to create the workstation machine, with the following changes:

- A different machine name (e.g. fileServer)
- The machine type is again Server
- the operating system type can be any server OS (e.g. Microsoft Windows Server 2003, any Linux distribution, etc.)

The fileServer machine now needs a user account. Create a root account on fileServer in the same way as on webServer previously.

Now, a file sharing service (Samba) will be added to the fileServer machine.

- Bring up the fileServer machine's context menu
- Select *New Service*
- Give the service a name (e.g. Samba)
- Use the *Run As* dropdown to select the root account previously created
- Enter the CPE for the file sharing service (*cpe:/a:samba:samba:3.0.27::*)
- Enter the port number that the service runs on (e.g. 139)
- Enter the name of the protocol used by the service (i.e. smb)
- Use the Type dropdown to select the *fileshare* type for this service

Note: *File shares based on user privileges (e.g. Windows shares) are not supported in this version of NetworkModeler.*

The fileServer machine now needs to have two file shares defined to accompany the Samba service. First, an export to the workStation machine:

- Bring up the fileServer machine's context menu
- Select *New Share* (which is available due to a *(*fileshare) type service being present on this machine)
- Give the file share a name (e.g. workStationFiles)
- Enter the local directory that is to be shared from the fileServer machine (e.g. /export/files)
- Use the *Access* dropdown to select the access type for this share (*all*)

- In the *Export To* field, enter the fully qualified name of the machine that will have access to this file share (e.g. 3host_network_workStation)

Next, an export to the webServer machine. Follow the same steps as above, with the following changes,

- The file share must have a different name (e.g. webServerFiles)
- The local directory should be */export/www*
- The access permissions should be *all*
- In the *Export To* field, enter the fully qualified name of the machine that will have access to this share is 3host_network_webServer

The fileServer machine is now complete. At this point, the three hosts for the sample network are complete. Now, an Internet zone with an attacker's machine will complete the topology of the scenario. Repeat the steps for creating a zone with an appropriate name (e.g. internet), and then create a machine of type Workstation within that zone for the attacker (e.g. attacker). This machine does not need to have any software or services installed on it.

5.1.4 Adding ACL Entries

The final step in the creation of the model is to add the appropriate ACLs for connectivity within the model. For this tutorial model, two ACLs will be needed:

- An ACL to indicate connectivity from the network zone to the internet zone
- An ACL to indicate connectivity from the internet zone to the webServer machine

To create the ACL for connectivity from the network to the internet:

- Bring up the context menu for the internet zone
- Select *New ACL*
- Give the ACL a name (e.g. network_internetACL)
- Set the ACL type to *full* using the Type dropdown
- Fill in the fully qualified name of the element from which the ACL originates in the From field- in this case, 3host_network

- Select the element type that the From field represents (zone or machine) - in this case, zone
- Enter the port(s) that the ACL allows communication over (see below). If the type is *full*, this field should be *any* (this is the default if left blank)
- Enter the protocol(s) that the ACL allows communication via (see below). If the type is *full*, this field should be *any* (this is the default if left blank)

Note that for the ACL type, there are three options:

1. single - used when describing a single port and protocol pair that is permitted by the ACL (e.g. 80 and http)
2. range - used when describing a set of port and protocol pairs that are permitted by the ACL. The format for the ports and protocols is comma separated lists (e.g. 80,25 and http,smtp)
3. full - used when describing an ACL that allows all port and protocol pairs

To create the ACL for connectivity from the internet zone to the webServer machine:

- Bring up the context menu for the webServer machine
- Select *New ACL*
- Give the ACL a name (e.g. internet_webServerACL)
- Set the ACL type to *single* using the Type dropdown
- Fill in the fully qualified name of the element from which the ACL originates in the From field - in this case, 3host_internet
- Select the element type that the From field represents (zone or machine) - in this case, zone
- Enter the port that the ACL allows communication over (i.e. 80)
- Enter the protocol(s) that the ACL allows communication via (i.e. http)

Now would be a good time to perform a save on the model.

5.1.5 Using Connectivity Walk to Verify ACLs

After the necessary ACLs are defined for a model, there is a mechanism by which their correctness may be verified. Using the *Model* → *Connectivity Walk* menu item, NetworkModeler can be placed in connectivity walk mode. In this mode, machines on the network can be selected and arrows will be drawn to show the available connections between the selected machine and other machines in the network. In connectivity walk mode for the tutorial network, the following connectivities should be observed:

- 3host_internet_attacker → 3host_network_webServer (80, http)
- 3host_network_fileServer → 3host_network_webServer (anyPort, anyProtocol)
- 3host_network_fileServer → 3host_network_workstation (anyPort, anyProtocol)
- 3host_network_fileServer → 3host_internet_attacker(anyPort, anyProtocol)
- 3host_network_webServer → 3host_network_fileServer (anyPort, anyProtocol)
- 3host_network_webServer → 3host_network_workstation (anyPort, anyProtocol)
- 3host_network_webServer → 3host_internet_attacker(anyPort, anyProtocol)
- 3host_network_workstation → 3host_network_fileServer (anyPort, anyProtocol)
- 3host_network_workstation → 3host_network_webServer (anyPort, anyProtocol)
- 3host_network_workStation→ 3host_internet_attacker(anyPort, anyProtocol)

To leave connectivity walk mode, use the *Model* → *Connectivity Walk* menu item to toggle it off.

5.1.5.1 Using the CPE Builder

NetworkModeler uses the Common Platform Enumeration (CPE) standard for identifying server and software elements, and hardware platforms (see http://cpe.mitre.org). The CPE specification defines a structured naming scheme to ensure conformity when describing a product.

The CPE builder included in NetworkModeler is a means for users to quickly build CPE identifiers for elements within networks. On every dialog where a CPE can be entered, there is a ... button to the right side. Click this button to launch the CPE builder. When the CPE builder appears, note the dropdowns present for each element shown in Figure 1,

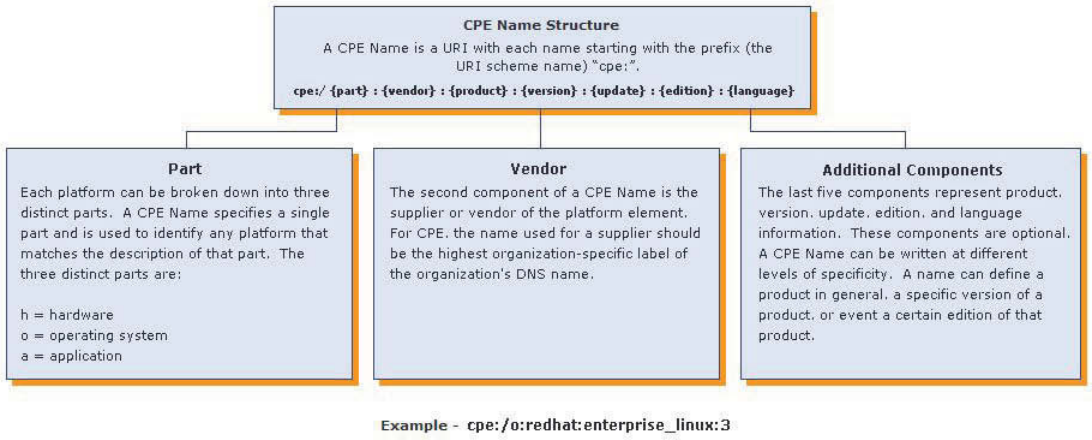

Figure 1: *CPE Naming Scheme*

with the exception of language. Simply start at the top dropdown and begin specifying the appropriate element of the CPE name for the entity being described. Each time a new element is selected, the next dropdown will be automatically populated with the available choices based on previous selections. For example, when the product element is selected, the version dropdown will be populated with all known versions of that product.

Note: *When a vendor is selected, the product dropdown will be populated with all known products available from that vendor that also match the part type that has been specified.*

The CPE builder depends on the CPE dictionary, which is an XML file included with NetworkModeler. The CPE dictionary is not a static document, and it is updated relatively frequently. If the CPE for a desired item cannot be built with the included version of the CPE dictionary, the latest version can be found at http://nvd.nist.gov/cpe.cfm. Copy the latest version of the dictionary to a convenient place. Next, enter the */home/user/SPADE/NetworkModeler/cpe* directory, and create a symbolic link to the CPE dictionary called *cpe-dictionary.xml*. The next time NetworkModeler is launched, the new CPE dictionary will be loaded.

Note: *When NetworkModeler is rebuilt from the NetBeans environment, the build process will reset the symbolic link to the version of the CPE dictionary that is included in the NetworkModeler source. If you have updated the version of the CPE dictionary being used, you will need to recreate the symbolic link to the version of the CPE dictionary that you are using, after the build process has finished.*

If a desired item still cannot be built using the latest version of the CPE builder, simply enter the CPE manually in the text box, rather than using the CPE builder.

5.2 Using a Model to Create Attack Graphs Using MulVAL

Using the three host model that was created in the first tutorial, MulVAL will now be invoked to create an attack graph.

For this process, one extra file needs to be created, external to NetworkModeler: The MulVAL attack file. This file is used to define elements for the MulVAL predicate generation that are not part of the network model proper. At this time, the only element that is required in the file defines the attacker's location in the scenario. Below is the format for the file, showing the declaration for the attacker in the *3host* site, *internet* zone, *attacker* machine:

```
<?xml version="1.0" encoding="UTF-8"?>
<model xmlns:mulval="http://www.bell.ca/mulval" version="1.1">
  <mulval>
    <attacker site='3host' zone='internet' location='attacker' />
  </mulval>
</model>
```

Once the MulVAL attack file is created, all pieces are in place to create and invoke MulVAL to generate an attack graph:

- Verify that the model is complete and correct
- *Model* → *Generate MulVAL*
- When prompted, navigate to and select the MulVAL attack file created for this model
- *Model* → *Invoke MulVAL*
- When RapidMiner opens, there will be a process already set up with the MulVAL invoker present and configured
- Run the process (press F5 or click the blue arrow on the toolbar)
- Assuming no errors, close RapidMiner
- The attack graph files will have been created in the model directory

5.2.1 Loading and Examining an Attack Graph

Once RapidMiner has run and invoked MulVAL, the attack graph files will be present in the model's directory, assuming no errors were encountered. This section will describe how attack graphs are loaded and analysed in NetworkModeler.

First, load the attack graph:

- In NetworkModeler select *Model* $\rightarrow$ *Open Attack Graph*
- Click the ... button for the Arc file
- Navigate to the 10host model directory and select the ARCS.CSV file
- Click the ... button for the Vertex file
- select the VERTEX.CSV file
- Select OK

The attack graph will now load. Note that black and red circles have been drawn on machines in the model. Black circles indicate attacker locations (as defined in the MulVAL attack file), while red circles indicate machines where execCode nodes occur in the attack graph.

5.2.1.1 Finding an Attack Path

By selecting an execCode node (red circle) and bringing up the context menu, note the option for *Find shortest attack Path*. Selecting this option will draw the shortest path from the attacker node (black circle) to the selected node. Note the *Attack Graph* panel in the lower left corner of the NetworkModeler window; it will list the relevant lines from the attack graph results that the attacker takes advantage of to move from attackerLocated to execCode on the selected machine. (The Copy button can be used to copy the contents of the panel to the system clipboard for pasting into other documents.)

5.2.1.2 Walking an Attack From the Attacker Node

By selecting the attacker node and bringing up the context menu, note that there is now an option for *find next execCode*. Selecting this option will draw arrow(s) on the network indicating which machine(s) can be reached from the attacker's machine to get to an execCode. Selecting a machine at the end of an arrow and bringing up its context menu, the *find next execCode* option will be still present if there are further paths from the selected machine to other machines in the network to continue. As with finding the shortest attack path, above, the *Attack Graph* panel will detail the path followed with the relevant lines from the attack graph.

5.2.1.3 Applying AssetRank to an Attack Graph

Instead of invoking MulVAL alone to get an attack graph, RapidMiner can be invoked to load a process that will run MulVAL and then AssetRank in sequence on the model to generate an attack graph that also has asset rankings attached to the results. In this case, select *Model* → *Generate MulVAL* as normal. Then, rather than selecting *Model* → *Invoke MulVAL*, select *Model* → *Invoke AssetRank* instead. RapidMiner will be started, then run the process as before. When the attack graph is generated and ranked, assuming no errors, load the attack graph as before. Now, select *Model* → *Show Weights* to show the AssetRank rankings on nodes as applicable.

5.2.2 Using SPADE With Other Tools

Some users may wish to use the SPADE environment in with tools other than MulVAL/AssetRank (or in addition to those tools.) RapidMiner operators specific to these tools have been written for the SPADE environment, and using them as a template it is certainly possible to write other operators to invoke other tools. However, it is not completely necessary to do so. The SPADE environment includes a generic invoker operator that can be used in place of a custom operator for a given tool, and the RapidMiner application includes a command line operator that can be used similarly for simpler needs.

5.2.2.1 SPADE Generic Invoker

If it is desired to add additional stages after invoking MulVAL (or AssetRank after MulVAL), the SPADE Generic Invoker is likely the best choice. Within RapidMiner, right clicking on the *Root* node of a process brings up a context menu with a *New Operator* item on the menu. Hovering over the New Operator item brings up another menu of operator categories. The *DRDC* category contains the *Generic Invoker* operator. Selecting the Generic Invoker will add it to the end of the process chain.

Note: *This operator assumes that the operator coming before it in the process has an output of two ExampleSets. An ExampleSet is a data structure within RapidMiner that is used to hold multi-column data (such as the CSV format used by attack graph arc and vertex files.) In the case of the MulVAL Invoker and AssetRank Invoker, all data is packaged in ExampleSet format, the first for the arcs and the second for the vertices, which is passed on to the next operator in the process, if any.*

By using the Generic Invoker, the ExampleSets that are being presented by the MulVAL or AssetRank Invoker may be written out to CSV files of the user's choice before invoking the command line for the tool to be used. To do this, select the Generic Invoker in the process. There will be five parameters for the operator:

- write first - This parameter takes the file name that the ExampleSet containing the arcs will be written to

- write second - This parameter takes the file name that the ExampleSet containing the vertices will be written to
- commandline - This parameter takes the entire command line to invoke the desired tool
- read first - This parameter takes the file name that the arcs should be read from, after the desired tool has modified them
- read second- This parameter takes the file name that the vertices should be read from, after the desired tool has modified them

The Generic Invoker will package the arcs and vertices back into ExampleSets after reading them, ready to pass on to another operator in the process. For example, a process might begin by invoking MulVAL, then using the Generic Invoker to use the desired tool to modify the generated attack graph in some fashion. If the next operator in the process is then the AssetRank Invoker, then AssetRank would be run against the attack graph arcs and vertices modified by the desired tool. Another example might be invoking MulVAL and then AssetRank in sequence, and then using the Generic Invoker to modify the attack graph as modified by AssetRank as the final step.

Note: *that the MulVAL Invoker always begins by reading a MulVAL predicate file; it does not take ExampleSet data from any previous operator. Any modification of the MulVAL predicate file must end by writing the modified .P file to disk, which will then be read by the MulVAL invoker.*

5.2.2.2 RapidMiner CommandLine Operator

RapidMiner also includes a very simple *CommandLine Operator* which is found in the *Core* sub-menu of the *New Operator* menu item. It takes a single input parameter of a command line to be invoked. Any outputs or files created as a result of the command line invoked should be handled with other operators (e.g. *New Operator* $\rightarrow$ *IO* $\rightarrow$ *Examples* $\rightarrow$ *SimpleExampleSource* to read a CSV file into an ExampleSet) to serve as appropriate inputs for other operators in the process.

6 Future Work

At the conclusion of the effort to create this experimenter, several research directions to further the capability of this toolset are have been identified and are described in this section.

It is recommended that more research be undertaken to create an effective mechanism for modelling, representing and identifying the dependancies on component level elements of a software package. As an example, the following scenarios may be more effectively incorporated into the SPADE model with a better modelling approach:

- Applications that are part of and used by a specific operating system distribution
- Operating system elements that are used as an application/service (e.g. Active Directory / MS File Sharing)
- Application dependancies on applications (e.g. shockwave and web browsers)

A detailed review of the NVD, CVSS and CPE schemas as well as data sampling from these sources will identify other inter-components that could be effective incorporated into the SPADE toolset model. Similarly, the presence and use of AND / OR operators in the description of vulnerability conditions should be investigated for inclusion into the SPADE toolset.

There is a need to conduct a detailed review of the existing and new MulVAL predicates to ensure that there is a method to represent each logic statement within the model. One current omission is the representation of the *nfsMounted* predicate, that is, a statement identifying that a particular network share has been mounted by a specific machine. Additionally, the MulVAL project should should be periodically reviewed to ensure that there is complete coverage of the predicates within the SPADE toolset.

Attacker information is currently not modelled through the interface. A separate XML file that reflects attacker information, most notably attacker location, is currently created manually. The SPADE toolset should allow this information to be set via the interface and should generate the required files for input to the MulVAL process.

The interface requires some minor enhancements, but a specific set of needed improvements should be derived through a user interface requirements review session conducted with a representative set of analysts who will eventually become users of the system. One noted issue is that a user has to explicitly left click on an interface element to acquire that element's context before you right click it to obtain its properties. If the analyst does not do this, the toolset uses the context of the last selected element.

7 Requirements Progress Review

This section provides a status review on the development of the experimenter by reviewing progress against the solution requirements that were defined at the beginning of the development effort. Note that these requirements were defined for the completed solution are were not meant to be a definition of the target of completion of this stage of the SPADE project.

7.1 General Requirements

These requirements describe needed functionality that reflects the overall intent of the solution and goals to be achieved in the architectural design of the application.

No.	Description	Priority	Status
GR-1	The system must be modular and extensible.	1	Done
GR-2	The individual solution modules must be uniquely identifiable for configuration purposes.	1	Done
GR-3	The modules must *plug* into a dynamic and configurable framework to meet the needs of researchers and end-users alike.	1	Done
GR-4	Definitions for a given type of module must provide for the means of swapping one module out for another. For example, replacing AssetRank with another asset ranking module.	1	Done
GR-5	Standards based formats such as XML must be used and they must be extensible with an open and well-defined format.	1	Done
GR-6	All assets being examined must be stored in a Microsoft SQL Server database with an open XML interface whose output is human-readable and manipulable. The structures must be sufficiently dynamic to allow for inclusion of new asset values and variables.	1	Done
GR-7	For the purposes of generating models, the system must allow assets (physical, logical) to be evaluated individually or as groups. That is, is should be possible to cut-and-paste or template systems and groups of systems.	1	Done

No.	Description	Priority	Status
GR-8	The system must allow users to observe the state of the analysis as it unfolds, such as profiling and breakpointing. Breakpoints should be able to be set at the module-level.	1	Partial. It is possible to stop processing at any time and to prevent the system from progressing beyond a point of interest.
GR-9	The system must allow for flexible grouping of assets according to predefined *values*, such as time, location, and type. The XML schema must support this capability.	1	Not Done. These information elements are currently not modelled.
GR-10	The system must allow for the prioritization of assets.	1	Partial. Priority targets can be set.
GR-11	The system must allow for the creation of templates and baselines so as to allow for rapid creation of massive networks from basic components.	1	Done
GR-12	Users must be able to specify needed details associated with an attack, specifically the attacker's node location and initial access privileges.	1	Done through separate XML structure
GR-13	The system should attempt to re-use existing technologies so as to minimize the development effort.	2	Done
GR-14	The system should be as simple as possible while still providing the needed analytical capabilities in support of security research.	2	Done
GR-15	The system, though composed of multiple independent modules, must behave and be called like a single application. An approach that uses multiple executables, for example one for loading/manipulating network models and one that integrates with a solution such as RapidMiner would be acceptable.	2	Done

No.	Description	Priority	Status
GR-16	Modules should be able to be authored in any language so long as they link together to the larger system whole. The primary purpose of this demonstrator is to easily enable further research extensions, therefore, modules written under this contract should be written in Python, Java, C++, Visual Basic and Datalog as these represent the languages used by DRDC research staff.	2	Done
GR-17	Several robust and documented APIs should be in place to support modularity and external connections	2	Done. There is a simple API for calling external components.
GR-18	Well-documented and simple APIs should be provided that are compatible with in-use technologies such as C/C++, Python, Java, MatLab, and Prolog.	2	Done.
GR-19	The system should allow multiple analysis modes. For example, the user community may define the need for such analysis modules as sensitivity analysis, trajectory analysis, and what-if analysis.	2	Partial as AssetRank and MulVAL provide some of these capabilities.
GR-20	The client should be a desktop application rather than a web-only application.	2	Done
GR-21	The system should support result filtering (e.g. only show attack graphs that are affecting Internet Explorer) for ease of use and to support need-to-know.	2	Done
GR-22	Within the model, it should be possible to define an asset as anything in the network environment, including files, devices, machines, and network nodes.	2	Partial. Visibility to the system, service, software level, but not to the file level.
GR-23	The system should provide details regarding a vulnerability once prioritized, for example, what is the vulnerability and what remedial measures are appropriate to counter it.	2	Not done
GR-24	Virtual machines may be used for snapshot ability and transportability of the system. As such, the solution should be delivered and installable as a virtual machine.	2	Done

No.	Description	Priority	Status
GR-25	The system must be able to take a predefined network definition and generate an equivalent network model.	2	Done when the SPADE XML definition is used.
GR-26	The topologies must be extendable and initially annotated to the degree required by the algorithms with information on assets, safeguards, connectivity, addresses, applications, etc..	2	Done
GR-27	The system must support import of data from multiple external sources, reusing rather than creating entirely new data sources such as JNDMS, IDS, BPEL, and NVD.	3	Done for NVD
GR-28	External applications should be able to use the output from the system, that is, output should be in XML where users can transform/analyze it independently.	3	Done
GR-29	The system may be aware of business processes as policies to be enforced and use them in the context of prioritization.	3	Not done
GR-30	These components must be stored within a database for later re-use, extension, or instantiation.	3	Done
GR-31	The solution should support release as open source software by both following and restating open source licensing restrictions.	3	Done
GR-32	The system must be able to run in a distributed fashion, wherein the modules can operate within a single computer system or across a network.	4	Done
GR-33	The system should provide extended details regarding a vulnerability once prioritized and reported, for example, how is the vulnerability exposed and what policy violations apply that lead to the vulnerability.	4	Not done
GR-34	The system should be able to prioritize and correlate vulnerability/event data (i.e. examine events, extract attributes, correlate to vulnerabilities, prioritize the vulnerabilities). As this system is pro-active only, this level of vulnerability analysis is out of scope for this project.	5	Not done

No.	Description	Priority	Status
GR-35	The system should be able to auto-discover and create the necessary topology models.	5	Not done
GR-36	The system must have an algorithm that defines the ÒcoverageÓ provided by existing safeguards, preferably with a ranking of effectiveness.	5	Not done
GR-37	The system must fit into the JNDMS architecture.		N/A
GR-38	The system and the XML structures and formats must comprehend the passage of time, allowing for temporal logic to be embedded into the asset description, that is work factor, as well as the various structures that define how the system functions.		N/A

Table 1: General Requirements

7.2 Operational Requirements

These requirements describe elements of the application pertaining to the operational capabilities that will be available to the analyst.

No.	Description	Priority	Status
OR-1	A robust XML schema for defining network and node elements must be found or created.	1	Done
OR-2	The system must create detailed logs of the analysis steps and status (metadata) to facilitate correctness verification of models.	1	Partial. There are error logs and processing status logs for most transformations.
OR-3	The system should allow users to replay models, including what-if scenarios, to ensure same results for same inputs.	1	Done
OR-4	It must be possible to replace network models and analysis modules in support of analysis.	1	Done
OR-5	The system should allow the end-user to indicate which variables are considered important, especially for data import from external sources or for output and pre-filtering.	1	Not done

No.	Description	Priority	Status
OR-6	The system must allow for the creation of multihop analysis.	1	Done
OR-7	The system must allow for the data to be provided in an open format (XML) so as to allow for end-user unique ways of accessing, assessing, and navigating the results.	1	Done
OR-8	The system must be able to perform internal logging of all relevant actions performed by the end-user.	2	Not done
OR-9	The system must allow invocation from external applications for data exchange and manipulation.	2	Done
OR-10	The system should allow for algorithm unfolding traceability that allows the user to watch how an algorithm works against a given set of data. The system should allow for full inspection of the data structures pre/post algorithm execution, for example, to support debugging efforts.	2	Partial
OR-11	The system should be able to indicate whether or not an asset is accessible from a given node and the facts which enable the access to occur.	2	Done
OR-12	Exposures (or risks) must be highlighted as to where they originate and how they impact the network model.	2	Partial. True for vulnerabilities
OR-13	The system should be available across various platforms (PC, Mac, Unix) where there is support for the tools that will define the solution components such as RapidMiner.	3	Partial since delivered as virtual machine
OR-14	The system must maximize the use of the mouse, for example, the ability to right click to bring up contextual menus.	3	Done
OR-15	The architecture must allow for efficient comparison with known facts. i.e.: if the system states that a machine is not vulnerable, but it is known to be vulnerable, that should be easily compared.	3	Not done
OR-16	The system should be able to provide Course of Action (COA) results whenever it discovers a threat.	3	Not done

No.	Description	Priority	Status
OR-17	Reports must be creatable, editable, and exist that sufficiently reduce the amount of information flowing back to the operational end-user (i.e., there is simply too much information, it needs to be reduced into either digestible chunks or distilled down to relevant data only)	3	Done
OR-18	The system must allow invocation of external applications for data exchange and manipulation.	4	Done
OR-19	The system should allow choice as to its scan range/depth (e.g. backbone, WAN, Internet exposed systems, servers, workstations)	4	Partial
OR-20	There should be a way of linking the results of this system back into JNDMS.	4	Not done
OR-21	The system must be able to perform discovery (topology, nodes, content) of a network to which it is introduced.	5	Not done
OR-22	The system may be capable of performing diagnostics (e.g. detect misconfiguration in firewall, detect deviance in router configuration from NSA hardening guidelines)	5	Not done
OR-23	The system may be capable of observing historical inputs to make better judgements regarding events. For example. 29 of 30 of the same event in the past four hours were false positives, then lower the priority of the 31st event unless some other parameter is also changed.		N/A
OR-24	The system should allow for data and reports to be dumped to a web page.		N/A
OR-25	The system must be able to determine a threat against an asset and provide the likelihood.		N/A
OR-26	Detected events must be ranked as to relevance and reliability of event based on known safeguards, etc. within the system being modeled.		N/A
OR-27	False positive and false negative issues should be addressed in a logical and consistent manner.		N/A
OR-28	Discovered dangers/threats should result in programmable measures that are passive (email the security officer) or active (shutdown systems).		N/A

No.	Description	Priority	Status
OR-29	Correlation with historic precedence must be maintained so as to retain long-term state and leverage temporal logic.		N/A

Table 2: Operational Requirements

7.3 Configuration Requirements

These requirements describe elements of the application that pertain to configuration. This includes configuration of the application itself, as well as definition of network models in the chosen descriptor language.

No.	Description	Priority	Status
CR-1	System configuration files must be human readable and modifiable.	1	Done
CR-2	Configuration files must allow for configuration of system, indicating which modules to load and in which order including pre- and post-processing.	1	Done
CR-3	Changes made via GUI must be reflected in configuration files.	1	Partial. Some elements such as attacker location not reflected in the interface.
CR-4	The system should have facilities to make model creation fast and simple (e.g. templating of nodes, cut and paste, etc.)	1	Done
CR-5	The system should reflect changes made to the model via the GUI in configuration files as the changes are made. There should be file consistency checks on save. That is, if the original configuration file has been changed the user should be warned before results are saved. We do not need try to merge the configuration file changes with the online changes, just identify when to potential source of change (GUI and config file) may impact each other.	2	Not done
CR-6	Changes made to configuration files should be detected in an active session.	2	Not done

No.	Description	Priority	Status
CR-7	Reporting formats should be defined in configuration files that may be copied and modified.	3	Not done
CR-8	The system may allow diff type comparisons on model files and configuration files to provide ease of comparison to see what is different between scenarios.	3	Not done

Table 3: Configuration Requirements

7.4 User Interface Requirements

These requirements describe elements of the application that pertain to presentation. This includes what is displayed to the user, as well as how the user interacts with the application.

No.	Description	Priority	Status
IR-1	The GUI presented to the user must be clean and manipulable, with the ability to drill down into detail for individual elements.	1	Done
IR-2	User interface must allow outputs to show results in multiple useful formats (tables, graphs, charts, etc.).	1	Done
IR-3	The GUI must be able to provide analysts with a user-friendly interaction method.	1	Done
IR-4	The system must provide text-based and graphical interfaces so as to allow calls from scripts that allow it to be an element in a complete simulation.	2	Done
IR-5	The system should allow facilities for model creation via GUI in addition to configuration files.	2	Done
IR-6	The system should allow access (read, modify, save, load) to AssetRank parameters.	2	Partial. AssetRank parameters taken as input.
IR-7	The system should allow access (read, modify, save, load) to MulVAL predicates.	2	Partial. Most but not all predicates covered (e.g. attacker location)

No.	Description	Priority	Status
IR-8	The system may allow the user to change the model, that is, by halting processing at a module level and allowing the manipulation of the execution parameters of subsequent modules..	2	Done
IR-9	The system should allow, for appropriate parameters, the iteration over a range of values.	2	Done for some parameters
IR-10	Text descriptions, similar to Visio, along graph edges should be supported.	3	Done. Graph edges have textual information related to the network and attack paths.
IR-11	User interface should allow outputs to show results in multiple contextual formats, including network view, asset view, and operations view..	4	Done. Multiple views supported including network walking and MulVAL graph interpretation.

Table 4: Interface Requirements

7.5 Persistence Requirements

These requirements describe elements of the application that pertain to how data is kept in non-volatile forms (i.e. files written to disk, database, etc.) This includes what is stored, the formats data is stored in, and manipulation of those storage formats.

No.	Description	Priority	Status
PR-1	Data persistence must be provided by MSSQL.	1	Done
PR-2	Interactions with the data service must be via open standards such as XML/XQuery.	2	Done
PR-3	The system should allow saving model states: resume where you left off, reload an earlier version, define a common start point for what-if scenario modelling, etc.	2	Done
PR-4	The system should allow data export in multiple formats (e.g. CSV, XML, SQL).	2	Done

Table 5: Persistence Requirements

7.6 Documentation Requirements

These requirements describe elements of the application's documentation. This includes architecture and design documentation, installation and maintenance documentation, test data, test guides, and user guides.

No.	Description	Priority	Status
DR-1	The system must be delivered with sufficient documentation to support the installation, operation and maintenance of the solution post-delivery.	1	Partial. Detailed user guides should be addressed in a subsequent project phase.
DR-2	Documentation must make integration by external parties as simple as possible.	2	Done

Table 6: Documentation Requirements

This page intentionally left blank.

DOCUMENT CONTROL DATA

(Security classification of title, body of abstract and indexing annotation must be entered when document is classified)

1. ORIGINATOR (The name and address of the organization preparing the document. Organizations for whom the document was prepared, e.g. Centre sponsoring a contractor's report, or tasking agency, are entered in section 8.)

   Bell Canada
   160 Elgin St., 17th Floor, Ottawa, Ontario, K2P 2C4

2. SECURITY CLASSIFICATION (Overall security classification of the document including special warning terms if applicable.)

   UNCLASSIFIED
   (NON-CONTROLLED GOODS)
   DMC A
   REVIEW: GCEC June 2010

3. TITLE (The complete document title as indicated on the title page. Its classification should be indicated by the appropriate abbreviation (S, C or U) in parentheses after the title.)

   Security Posture Assessment Demonstrator and Experimenter

4. AUTHORS (Last name, followed by initials – ranks, titles, etc. not to be used.)

   Bacic, E.; Henderson, G.; Tremblay, L.

5. DATE OF PUBLICATION (Month and year of publication of document.)

   May 2011

6a. NO. OF PAGES (Total containing information. Include Annexes, Appendices, etc.)

   58

6b. NO. OF REFS (Total cited in document.)

   0

7. DESCRIPTIVE NOTES (The category of the document, e.g. technical report, technical note or memorandum. If appropriate, enter the type of report, e.g. interim, progress, summary, annual or final. Give the inclusive dates when a specific reporting period is covered.)

   Contract Report

8. SPONSORING ACTIVITY (The name of the department project office or laboratory sponsoring the research and development – include address.)

   Defence R&D Canada – Ottawa
   3701 Carling Avenue, Ottawa ON K1A 0Z4, Canada

9a. PROJECT OR GRANT NO. (If appropriate, the applicable research and development project or grant number under which the document was written. Please specify whether project or grant.)

   15BB03

9b. CONTRACT NO. (If appropriate, the applicable number under which the document was written.)

   W7714-071028

10a. ORIGINATOR'S DOCUMENT NUMBER (The official document number by which the document is identified by the originating activity. This number must be unique to this document.)

   DRDC Ottawa CR 2011-012

10b. OTHER DOCUMENT NO(s). (Any other numbers which may be assigned this document either by the originator or by the sponsor.)

11. DOCUMENT AVAILABILITY (Any limitations on further dissemination of the document, other than those imposed by security classification.)

   (X) Unlimited distribution
   ( ) Defence departments and defence contractors; further distribution only as approved
   ( ) Defence departments and Canadian defence contractors; further distribution only as approved
   ( ) Government departments and agencies; further distribution only as approved
   ( ) Defence departments; further distribution only as approved
   ( ) Other (please specify):

12. DOCUMENT ANNOUNCEMENT (Any limitation to the bibliographic announcement of this document. This will normally correspond to the Document Availability (11). However, where further distribution (beyond the audience specified in (11)) is possible, a wider announcement audience may be selected.)

13. ABSTRACT (A brief and factual summary of the document. It may also appear elsewhere in the body of the document itself. It is highly desirable that the abstract of classified documents be unclassified. Each paragraph of the abstract shall begin with an indication of the security classification of the information in the paragraph (unless the document itself is unclassified) represented as (S), (C), or (U). It is not necessary to include here abstracts in both official languages unless the text is bilingual.)

There is a recognized need within the network defence community for analysis tools that can assist in the evaluation of networks for vulnerabilities and aid in the design of robust networks. This paper provides a summary of the software development effort to create a pluggable software framework that can leverage network defence analysis tools, including advanced technical and research elements, in a unified framework in support of the design of secure and robust networks. The technology components for this framework, specifically RapidMiner and a series of solution specific software elements, are described in the context of the operational environment configuration in which these analysis tools are deployed. A high-order conceptual workflow is provided that details the processing logic, dependencies and data flow that constitute the operational practices of the network analysis framework. Build instructions for the creation and maintenance of the tool set are provided and a tutorial oriented toward the target user community describes how this analytical tool can be used to perform an assessment of a sample network and how to interpret the results using the integrated visualization capabilities. A series of suggestions for future enhancement of the tool set are similarly provided.

14. KEYWORDS, DESCRIPTORS or IDENTIFIERS (Technically meaningful terms or short phrases that characterize a document and could be helpful in cataloguing the document. They should be selected so that no security classification is required. Identifiers, such as equipment model designation, trade name, military project code name, geographic location may also be included. If possible keywords should be selected from a published thesaurus. e.g. Thesaurus of Engineering and Scientific Terms (TEST) and that thesaurus identified. If it is not possible to select indexing terms which are Unclassified, the classification of each should be indicated as with the title.)

attack graph; network model; AssetRank; MulVAL; RapidMiner; VMWare; Common Platform Enumeration CPE

Defence R&D Canada

Canada's leader in Defence
and National Security
Science and Technology

R & D pour la défense Canada

Chef de file au Canada en matière
de science et de technologie pour
la défense et la sécurité nationale

www.drdc-rddc.gc.ca