



**DN0678: 13 FEBRUARY 2006
ISSUE 4/2: 06 FEBRUARY 2012**

DESIGN DOCUMENT

FOR THE

**TECHNOLOGY DEMONSTRATION OF THE JOINT
NETWORK DEFENCE AND MANAGEMENT SYSTEM
(JNDMS) PROJECT**

PWGSC CONTRACT NO. W7714-040875/001/SV

CDRL/DID SD-004

**PREPARED FOR:
DEFENCE R&D CANADA - OTTAWA 3701 CARLING
AVENUE
OTTAWA ON K1A 0Z4**

DRDC Technical Authority: Maxwell Dondo, DS

**PREPARED BY:
SCOTT MACDONALD
MACDONALD DETTWILER AND ASSOCIATES LTD.
SUITE 60, 1000 WINDMILL RD.
DARTMOUTH NS B3B 1L7**

The scientific or technical validity of this Contract Report is entirely the responsibility of the Contractor and the contents do not necessarily have the approval or endorsement of the Department of National Defence of Canada.

**Contract Report
DRDC-RDDC-2015-C134
February 2012**

DOCUMENT APPROVAL SHEET

DESIGN DOCUMENT

**FOR THE
TECHNOLOGY DEMONSTRATION OF THE JOINT NETWORK
DEFENCE AND MANAGEMENT SYSTEM (JNDMS) PROJECT**

CONTRACT NO. W7714-040875/001/SV

CDRL /DID SD-004

MACDONALD DETTWILER AND ASSOCIATES LTD.

<u>Scott MacDonald</u> Author	_____ (Signature)	_____ (Date)
_____ Quality Assurance	_____ (Signature)	_____ (Date)
<u>Brett Trask</u> Project Manager	_____ (Signature)	_____ (Date)

TABLE OF CONTENTS

1	INTRODUCTION.....	1-1
1.1	OVERVIEW.....	1-1
1.2	SCOPE	1-1
1.3	REFERENCED DOCUMENTS	1-2
2	DESIGN OVERVIEW	2-1
2.1	AGENTS.....	2-3
2.2	COMPONENT DESCRIPTIONS.....	2-3
2.2.1	<i>JNDMS Services (JSS)</i>	2-3
2.2.2	<i>Decision Support System (DSS)</i>	2-4
2.2.3	<i>JNDMS Data Warehouse (JDW)</i>	2-4
2.2.4	<i>JNDMS User Interface (JUI)</i>	2-4
2.2.5	<i>Enterprise Infrastructure Management (EIM)</i>	2-5
2.2.6	<i>Security Information Management (SIM)</i>	2-5
2.2.7	<i>Situational Awareness Data Sharing</i>	2-5
2.2.8	<i>Vulnerability Assessment (VA)</i>	2-6
2.2.9	<i>Military Operations Data</i>	2-6
2.2.10	<i>Vulnerability and Exploit Data</i>	2-6
2.2.11	<i>Safeguard Data</i>	2-7
2.2.12	<i>Trouble Tickets (TT)</i>	2-7
2.2.13	<i>Security Sensors</i>	2-7
2.2.14	<i>Telecom Circuit Data</i>	2-7
2.2.15	<i>IT Infrastructure (Managed Base)</i>	2-8
2.2.16	<i>Data Transformation</i>	2-8
2.2.17	<i>Map Server</i>	2-8
2.3	INTERFACE SUMMARY	2-9
2.4	DEVELOPMENT ENVIRONMENT	2-14
2.5	SYSTEM SECURITY OVERVIEW.....	2-17
2.6	MODEL RELATIONSHIPS.....	2-17
2.6.1	<i>Primary Entities</i>	2-17
2.6.2	<i>Secondary Entities</i>	2-18
2.6.3	<i>Entity Relationships</i>	2-18
2.6.4	<i>Asset links</i>	2-23
3	SUBSYSTEM DESIGN	3-1
3.1	SYSTEM INPUTS	3-2
3.1.1	<i>Terminology</i>	3-2
3.1.2	<i>IT Infrastructure Data</i>	3-4
3.1.3	<i>Military Operations Data</i>	3-6
3.1.4	<i>Vulnerability and Exploit Data</i>	3-8
3.1.5	<i>Safeguards Data</i>	3-17
3.1.6	<i>Security Events</i>	3-31
3.2	ENTERPRISE INFRASTRUCTURE MANAGEMENT	3-34
3.2.1	<i>Overview</i>	3-34
3.2.2	<i>COTS Selection</i>	3-34
3.2.3	<i>System Interfaces</i>	3-36
3.2.4	<i>Subsystem Interfaces</i>	3-37
3.2.5	<i>Data Storage</i>	3-38
3.2.6	<i>Rules</i>	3-39
3.2.7	<i>Integration and Transition</i>	3-39

3.3	SECURITY INFORMATION MANAGEMENT	3-40
3.3.1	Overview.....	3-40
3.3.2	COTS Selection.....	3-41
3.3.3	System Interfaces.....	3-43
3.3.4	Subsystem Interfaces.....	3-46
3.3.5	Data Storage.....	3-49
3.3.6	Rules / Data Reduction.....	3-50
3.3.7	ISM Alarm Escalations to JNDMS.....	3-53
3.3.8	Scenarios.....	3-71
3.4	DATA TRANSFORMATION	3-73
3.4.1	Overview.....	3-73
3.4.2	COTS Selection.....	3-73
3.4.3	JSS Client.....	3-75
3.5	JNDMS DATA WAREHOUSE.....	3-79
3.5.1	Component Introduction.....	3-79
3.5.2	Component Description.....	3-79
3.5.3	Component Technologies.....	3-80
3.5.4	Data Model.....	3-85
3.6	JNDMS SERVICES.....	3-279
3.6.1	Overview.....	3-279
3.6.2	Interfaces.....	3-280
3.6.3	JSS Internals.....	3-296
3.7	DECISION SUPPORT SYSTEM (DSS)	3-299
3.7.1	Overview.....	3-299
3.7.2	COTS Selection.....	3-299
3.7.3	Subsystem Interfaces.....	3-301
3.7.4	Data Storage.....	3-302
3.7.5	Situational Awareness model.....	3-303
3.7.6	Overview of DSS processing stages.....	3-320
3.7.7	Initialization.....	3-321
3.7.8	Event Processing.....	3-326
3.8	PRESENTATION, VISUALIZATION AND ALERTING	3-328
3.8.1	Overview.....	3-328
3.8.2	COTS Selection.....	3-328
3.8.3	Component Design.....	3-329
3.8.4	Subsystem Interfaces.....	3-346
3.8.5	User Interface / Portal Design.....	3-346
3.8.6	Visualization Applet.....	3-372
3.9	SITUATIONAL AWARENESS DATA SHARING	3-375
3.9.1	Setting up data sharing.....	3-375
3.9.2	Sharing example configuration.....	3-377
APPENDIX A	ACRONYMS.....	A-1
APPENDIX B	CND DATA MODEL	B-1
APPENDIX C	SAMPLE SAFEGUARD DATA INPUTS.....	C-1
APPENDIX D	SAMPLE OPERATIONS DATA.....	D-1
APPENDIX E	SAMPLE VULNERABILITY REPORT	E-1
APPENDIX F	JSS INTERFACE DEFINITION (WSDL).....	F-1
APPENDIX G	EXECUTION ENVIRONMENT	G-1

LIST OF FIGURES

Figure 2-1: JNDMS Design Overview	2-2
Figure 2-2: JNDMS Physical Environment.....	2-14
Figure 2-3: Connections Between Assets.....	2-23
Figure 3-1: JNDMS Subsystems	3-1
Figure 3-2: Spectrum and Unicenter Communications	3-37
Figure 3-3: Designing the Data Warehouse from an Entity-Relationship Data Model .	3-85
Figure 3-4: JSS Overview.....	3-279
Figure 3-5: JSS threads and queues	3-296
Figure 3-6: Situational Awareness Model	3-303
Figure 3-7: MARPAC Ops Service Dependencies.....	3-311
Figure 3-8: IT Topology Model	3-322
Figure 3-9: Service Dependency Model.....	3-323
Figure 3-10: Service Paths Graphical Representation	3-324
Figure 3-11: GIS Information Flow.....	3-330
Figure 3-12: Portal Layout.....	3-347
Figure 3-13: Login screen.....	3-348
Figure 3-14: Layout options for JGraph: “Compact Tree” (top) and “Organic” (bottom)...	3-372
Figure 3-15: JGraph view port widget, including highlighted asset at risk	3-373
Figure 3-16: Zoomed-out visualization with zoom context menu, including link to detail page.....	3-373

LIST OF TABLES

Table 1: System Interface Summary	2-9
Table 2: Sensor Interface Summary	2-13
Table 3: Cycle 3 Workstation Configuration.....	2-15
Table 4: Asset Count by Type and Category in Development Dataset	2-26
Table 5: Operational Attributes from IAT	3-6
Table 6: Vulnerability Information from the NVD.....	3-10
Table 7: Vulnerability and Exploit Data Available in IAT	3-12
Table 8: Nessus Scan	3-15
Table 9: Common Safeguards.....	3-19
Table 10: Possible Sources of Security Events	3-32
Table 11: Trouble Ticket Attributes.....	3-33
Table 12: EIM Component Roles.....	3-34
Table 13: Security Feature Categories	3-47
Table 14: JSS Client Operations	3-75
Table 15: JNDMS Database Users.....	3-84
Table 16: JSS Data Inputs.....	3-282
Table 17: List of JNDMS Incident Types.....	3-318

1 Introduction

1.1 Overview

The purpose of this document is to specify the Joint Network Defence and Management System (JNDMS) design and satisfy the Contract Data Requirements List (CDRL)/Data Item Description (DID) SD-004 deliverable as stated in Contract No. W7714-040875/001/SV.

This design document represents a 'live' document with an initial release during Phase 1 and other releases at the end of cycle 2 and cycle 3 development of the JNDMS Technology Demonstration (TD). This release represents the design as documented at the end of Phase 3.

1.2 Scope

The document represents the system design of the JNDMS. This document extends the JNDMS architecture and provides the details on components, their interactions and their interfaces.

1.3 Referenced Documents

- [R-1] Request for Proposal (RFP) for the Technology Demonstration of the JNDMS, Solicitation No. W7714-4-040875/A, Public Works and Government Services Canada (PWGSC).
- [R-2] Proposal for the Technology Demonstration of the JNDMS, Proposal No. 01-4023, Revision 1/0 dated 09 June 2005, MDA.
- [R-3] Architecture Design Document for the Technology Demonstration of the JNDMS Project, DN0654, Issue 1/0 dated 21 October 2005, MDA.
- [R-4] System Requirements Specification for the Technology Demonstration of the JNDMS Project, DN0665, Issue 2/0 dated 17 January 2006, MDA.
- [R-5] Computer Network Defence (CND) Situation Awareness (SA). Draft for Comment, Julie Lefebvre, Marc Grégoire, Luc Beaudoin, and Michael Froh, Defence R&D Canada (DRDC).
- [R-6] Information Requirements Impact Assessment Tool for CND, A Canadian Forces perspective. Mr. Marc Grégoire, DRDC Ottawa, Luc Beaudoin, Synapse Connexion Inc, Technical Memorandum, DRDC Ottawa TM-2004-XXX, 2004-06-XX (draft).
- [R-7] Common Vulnerability Score System (CVSS), Final Report and Recommendations by the Council. John T. Chambers, John W. Thompson, October 12, 2004, National Infrastructure Advisory Council.
- [R-8] Taxonomy of the Computer Security Incident related terminology. TERENA Incident Taxonomy and Description Working Group, http://www.terena.nl/activities/tf-csirt/iodef/docs/i-taxonomy_terms.html
- [R-9] The Information Assurance Technical Framework (IATF) document, Release 3.1 dated September 2002.
- [R-10] Threat and Risk assessment Working Guide (CSE Method).
- [R-11] Security Orders for Classified Information Systems, D IM Secure, March 1, 2002, Department of National Defence.

2 Design Overview

The JNDMS design as seen in Figure 2-1 represents a number of Commercial Off-The-Shelf (COTS) products integrated into a system with several custom components. This design concentrates on defining the interfaces between components and allows flexibility in the choice of languages to accomplish the various tasks.

The core of the JNDMS is based on Java components that provide the key JNDMS custom components for the Decision Support System (DSS), the JNDMS Services (JSS) and the presentation components (JUI). The protocols and the data formats defined for the interfaces allow other languages to be chosen for components that interact with the JNDMS.

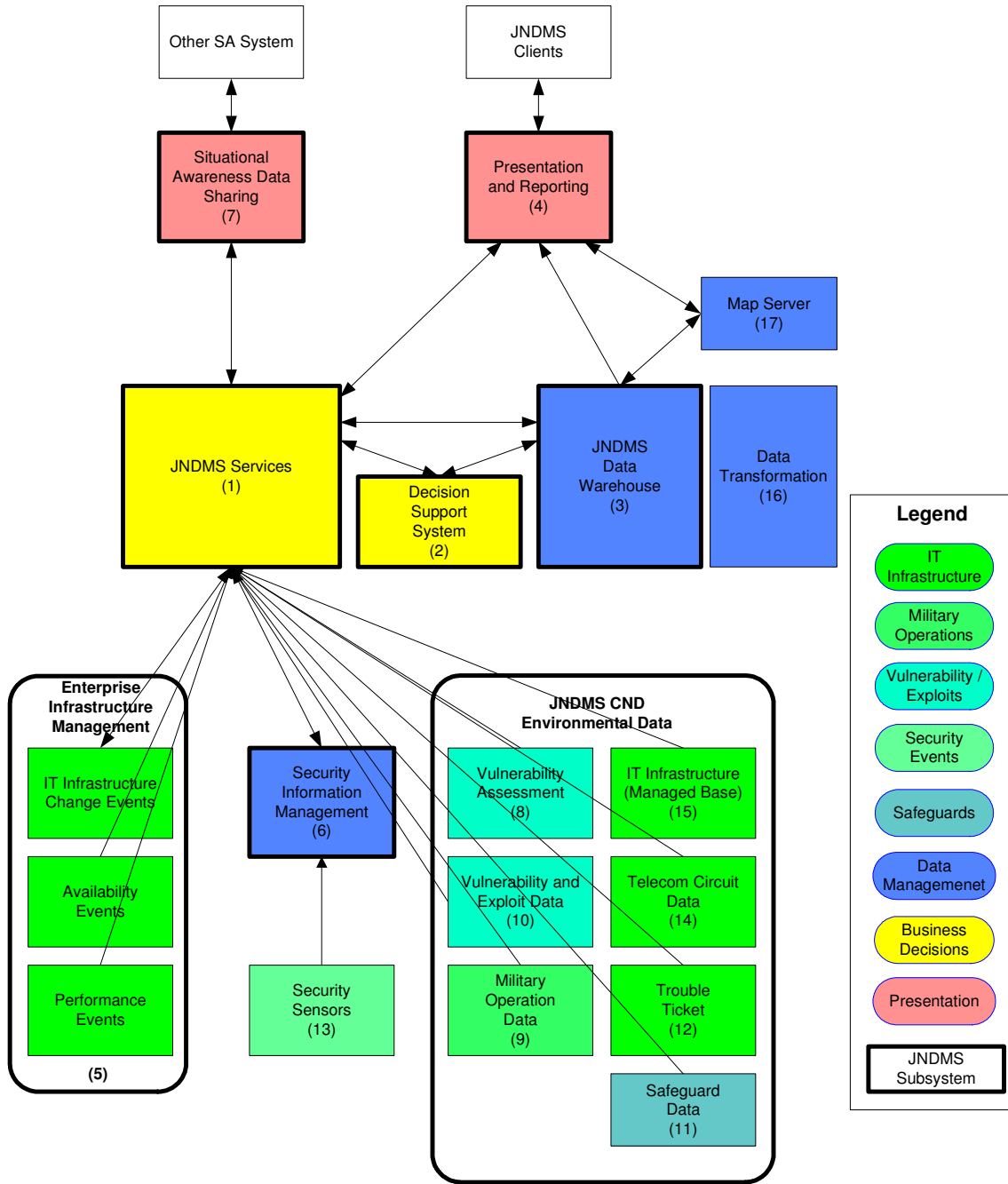


Figure 2-1: JNDMS Design Overview

2.1 Agents

The need to deploy agents across the network may cause issues during transition, may reduce the effectiveness of a JNDMS in many scenarios, but can also provide key information that is not available by any other method. The general guideline that will be used during development is that as much information as possible will be gathered with the following priorities:

1. No expectations of installed agents
2. Reliance on standard agents such as SNMP (Simple Network Management Protocol) MIBs (Management Information Bases)
3. Reliance on installed software or agents

Development will use all three approaches to evaluate the data flow and integration.

The deployment activities during the progression of the project found that in many cases, although agents themselves may not be required, at least administrative privileges must be granted. Without some elevated permissions for some of the core enterprise tools, the collection of software inventory as well as complete vulnerability information was only partial. The building of the core infrastructure also required access to SNMP communities, especially on core routers.

2.2 Component Descriptions

Figure 2-1 shows the components within JNDMS and shows the data flow between each of them. The following sections provide a brief description of each component with details available in Section 3 (Subsystem Design).

2.2.1 JNDMS Services (JSS)

Purpose: The JNDMS Services component provides data normalization and brokering between other JNDMS subsystems. This component is responsible for providing a web service interface for other JNDMS components, and external systems to interact with the JNDMS.

Interfaces: a) Simple Object Access Protocol (SOAP)

Java Database Connectivity (JDBC)

Technology: Tomcat Application Server

2.2.2 Decision Support System (DSS)

Purpose: The JNDMS Decision Support System (DSS) component will identify security threats and events (incidents and vulnerability instances) and will assign the appropriate level of severity to them.

The role of the DSS is to provide the impact on Situational Awareness (SA) of all of the events and incidents. The inputs to the DSS may be pre-processed events such as these provided by the SIM.

Interfaces: a) Java API

Technology: Custom Java Application
Optional Computer Associates (CA) CleverPath Aion Business Rules Engine (BRE)

2.2.3 JNDMS Data Warehouse (JDW)

Purpose: The JNDMS Data Warehouse (JDW) is a repository of information accumulated from a variety of sources. Prior to being stored in the JDW, this multi-source data will be transformed, and then aggregated so that data from different sources gets stored in a single repository.

Interfaces: a) Java Database Connectivity (JDBC)
b) Open Database Connectivity (ODBC)

Technology: Oracle Enterprise Server 10g / 11g.

2.2.4 JNDMS User Interface (JUI)

Purpose: The JNDMS User Interface component provides a situational awareness web portal to JNDMS users.

Interfaces: a) W3C Web Standards
b) Keyhole Markup Language (KML)
c) WMS

Technology: Google Web Toolkit (GWT)
Ext GWT (extended components for GWT)
Java
Google Earth Plugin

Open Layers Mapping Client

2.2.5 Enterprise Infrastructure Management (EIM)

Purpose: The Enterprise Infrastructure Management (EIM) component monitors the availability and performance of Information Technology (IT) assets. EIM also scans the network infrastructure to discover and map connected IT assets.

Interfaces:

- a) Java Database Connectivity (JDBC)
- b) Simple Object Access Protocol (SOAP)
- c) Command Line Interface (CLI)
- d) Various – interactions with IT assets and Services

Technology: CA Spectrum
Optional CA Unicenter Network and System Management (NSM)
Centennial

2.2.6 Security Information Management (SIM)

Purpose: The Security Information Management (SIM) component will process data inputs from the CND environment. From this data stream, significant security events and other events of special interest will be discovered.

Interfaces:

- a) Simple Object Access Protocol (SOAP)
- b) Command Line Interface (CLI)
- c) Various – see Security Sensor Data

Technology: Intellitactics Security Manager (ISM)

2.2.7 Situational Awareness Data Sharing

Purpose: The Situational Awareness Inputs component shares relevant and permissible situational awareness status and incidents with other situational awareness systems, including other JNDMS instances.

Interfaces: a) Simple Object Access Protocol (SOAP)

Technology: Custom software

2.2.8 Vulnerability Assessment (VA)

Purpose: The Vulnerability Assessment (VA) component scans IT assets to discover vulnerability instances.

Interfaces: a) Simple Object Access Protocol (SOAP)

Technology: Nessus Vulnerability Scanner
nCircle IP360

2.2.9 Military Operations Data

Purpose: The Military Operations Data component is a description of military operations and their dependence on specific IT assets. The Military Operations Data captures the IT services requirements for all domestic, international or deployed operations.

Interfaces: a) Java Database Connectivity (JDBC)
b) Simple Object Access Protocol (SOAP)

Technology: Simulated data
C2IEDM Data Sources (simulated for the TD)

2.2.10 Vulnerability and Exploit Data

Purpose: The Vulnerability and Exploit Data component integrates known cyber vulnerability definitions at the disposal of the organization, physical vulnerabilities and the known exploitation methods.

Interfaces: a) Web Services
b) Real Simple Syndication (RSS)
c) Simple Object Access Protocol (SOAP)

Technology: National Vulnerability Database (NVD)
Simulated data

2.2.11 Safeguard Data

Purpose: The Safeguard Data component describes the protection devices, detection devices or security features of assets.

Interfaces: a) Java Database Connectivity (JDBC)
b) Simple Object Access Protocol (SOAP)

Technology: Simulated data

2.2.12 Trouble Tickets (TT)

Purpose: The Trouble Ticket (TT) component harvest recent trouble ticket information from the organization's trouble reporting ticket.

Interfaces: a) Java Database Connectivity (JDBC)
b) Simple Object Access Protocol (SOAP)

Technology: Simulated data.

2.2.13 Security Sensors

Purpose: Security Sensor Data component represents the events gathered from security point products such as firewalls, intrusion detection systems (IDS), and operating system logs.

Interfaces: a) Various

Technology: Various including Snort IDS, Cisco IDS, Checkpoint firewall.

2.2.14 Telecom Circuit Data

Purpose: The Telecom Circuit Information component provides information regarding Telco supplied circuits in use within the Department of National Defence (DND) networks.

Interfaces: a) Java Database Connectivity (JDBC)
b) Simple Object Access Protocol (SOAP)

Technology: Simulated data

2.2.15 IT Infrastructure (Managed Base)

- Purpose: The Infrastructure (Managed Base) component provides a snapshot of the discovered IT infrastructure baseline.
- Interfaces: a) Java Database Connectivity (JDBC)
b) Custom event application using SOAP
- Technology: Simulated data from DND Configuration Management Database (CMDB).

2.2.16 Data Transformation

- Purpose: The Data Transformation component represents tools and interfaces to perform data transformation tasks.
- Interfaces: a) Various defined by the tools including ODBC, JDBC, Ibatis and eXtensible Markup Language (XML)
- Technology: XSLT Java language features
Custom transformations

2.2.17 Map Server

- Purpose: The Map Server is responsible for providing GIS related information to JNDMS. This component is responsible for GIS related queries and providing both graphical and text responses.
- Interfaces: a) Web interface (W3C Standards)
b) Database access (JDBC)
c) KML
- Technology: The final build of JNDMS relied on external map servers to provide WMS map layers or to provide the map data for Google Earth. Additional geographical information was provided by custom software to create KML content.

2.3 Interface Summary

Table 1 summarizes the interfaces between components. The “information/component” column shows what information or what JNDMS component the interface refers to. The “source” column identifies the specific tool, information source or even subsystem from which the information will be gathered. The “interface” column identifies how the information will be gathered from the source. The “format” column identifies how the information is formatted as it is gathered from the source. The “agent notes” column describes how the information will be transformed from the source’s interface and format into a format suitable for JNDMS. The agents, in this sense, are any supporting tools used to manage the transformation.

Table 1: System Interface Summary

ID	Information / Component	Source	Interface	Format	Agent Notes	Final Build	Design Section
1	JNDMS Services	JNDMS Component	SOAP over HTTP	XML	N/A	Y	3.5
2	DSS Rules	AION BRE	Aion API	API	N/A	N	3.7
2	DSS	JNDMS Component	Java API	API	N/A	Y	3.7
3	Data Warehouse	Oracle	JDBC	API	N/A	Y	3.5
4	JNDMS User Interface	Tomcat Server / GWT	Web Standards	Web Standards	N/A	Y	3.8
5	IT Infrastructure Change Events	Unicenter Network and Systems Management Centennial Discovery	NSM Event Interface,	Custom ASCII,	JSS Client	N	3.2
6	Software Inventory/Security Information Management	Centennial Discovery	Centennial database	JDBC	JSS Client	Y	3.2
7	IT Infrastructure Availability Event Data/Situational Awareness Data Sharing	Unicenter Network and Systems Management, Spectrum	NSM Event Interface	Custom ASCII	JSS Client	Y (Spectrum)	3.2
8	IT Infrastructure Performance Data/Vulnerability Assessment	eHealth, Spectrum	NSM Event Interface	Custom ASCII	An agent will translate from ASCII to SOAP, JSS Client	N	3.2

DN0678
Issue 4/2: 06 February 2012

ID	Information / Component	Source	Interface	Format	Agent Notes	Final Build	Design Section
6	SIM security event	SIM	SIM Extension SOAP over HTTP	SOAP XML	SIM can be extended with Java to export SOAP events, JSS Client	Y	3.3
7	Other Situational Awareness Data (coalition)	(simulated)	SOAP over HTTP	XML	No specific sources of external SA data are currently known. Samples will be simulated.	Y	3.9
7	Other Situational Awareness Data (JNDMS)	JNDMS	SOAP over HTTP	XML	None required.	Y	3.9
7	Data Sharing/Situational Awareness Data Sharing	JNDMS Component	N/A	As required	No specific sharing targets have been identified. This component will make queries of the JNDMS services and translate.	Y	3.9
8	Vulnerability Instances/Vulnerability Assessment	eTrust Vulnerability Manager	eTrust Reporting	CSV	SIM DMFD.	N	3.1
8	Vulnerability Instances/Vulnerability Assessment	Nessus Vulnerability Scanner	SCP	NBE	SIM DMFD. This is a push from the Nessus host. Requires a script.	Y	3.1
8	Vulnerability Instances/Vulnerability Assessment	nCircle IP360	SIH (Security Intelligence Hub)	JDBC	An agent will poll the SIH and translate into SOAP.	Y	3.1

DN0678
Issue 4/2: 06 February 2012

ID	Information / Component	Source	Interface	Format	Agent Notes	Final Build	Design Section
9	Military Operations Data	IAT	MS Access	ODBC	Operation data from IAT can be manually transferred to JNDMS. Snapshots of this data will be used for demonstrations.	Y (manual import)	3.1
9	Military Operations Data	(simulated from C2IEDM sources)	SOAP	XML	Simulated C2IEDM source can be used.	Y	3.1
10	Vulnerability and Exploit data	IAT	MS Access	ODBC	Vulnerability data can be manually imported into JNDMS for demonstration.	Y (manual import)	3.1
10	Vulnerability and Exploit data (CVE)	NVD	RSS/HTTP	XML	An agent will translate the RSS or HTTP feed into SOAP calls.	Y	3.1
11	Safeguard data	IAT	Access DB	ODBC	Safeguard data can be manually extracted from IAT for demonstration purposes. Some safeguard information will be configured within the JNDMS.	Y (manual import)	3.1
11	Safeguard data	Simulated	(simulated)	XML	Safeguard data can be simulated.	Y	3.1
12	Trouble Tickets	Ticket System	(simulated)	XML	Simulated trouble tickets submitted directly to JNDMS services.	Y	3.7

DN0678
Issue 4/2: 06 February 2012

ID	Information / Component	Source	Interface	Format	Agent Notes	Final Build	Design Section
14	Telecon Infrastructure/Telecom Circuit Data	DND NTMS	Oracle database	N/A	This will be simulated for the TDP.	N	3.1
15	IT Infrastructure (Managed Base)	DND CMDB	(simulated)	(direct to database)	This will be a snapshot of the discovered baseline and will be loaded directly into the data warehouse.	N	3.2
16	Data Transformation	JNDMS Component	N/A	N/A	This represents a number of Application Programming Interfaces (API) and tools to support agent development and transformations.	Y	3.4

Table 2 shows the sensors that flow into the JNDMS.

Table 2: Sensor Interface Summary

ID	Sensor	Source	Interface	Format	Agent Notes	Section
1	Sensor: Web application logs	Web server / reverse proxy	SCP	W3C Logs	SIM DMFD. This is a push from the Web server. Requires a script.	3.3
2	Sensor: Cisco IDS Alert	Cisco IDS sensor	RDEP	XML	SIM DMFD	3.3
3	Sensor: Firewall Logs	Pix Firewall	Syslog	ASCII	SIM DMFD	3.3
4	Sensor: Firewall Logs	Checkpoint FW-NG	Checkpoint LEA	Proprietary	SIM DMFD	3.3
5	Sensor: Access Control Policy Event	Gateway/router	Syslog	ASCII	SIM DMFD	3.3
6	Sensor: OS events	Windows OS Events	Windows Event Log	Proprietary	NSM Event Handling will export in ASCII, then an agent will translate into SOAP.	3.2
7	Sensor: OS Events	Unix OS Events	Syslog	ASCII	NSM Event Handling will export in ASCII, then an agent will translate into SOAP.	3.2
8	Sensor: OS Security Events	Unix OS Events	Syslog	ASCII	SIM DMFD	3.3
9	Sensor: Snort IDS Alert	Snort IDS sensor	Syslog	ASCII	SIM DMFD	3.3

2.4 Development Environment

The JNDMS development environment makes use of several operating systems, as well as virtual environments, to host the core of the system, the testing environment and the COTS selected to demonstrate the system. Figure 2-2 shows the configuration.

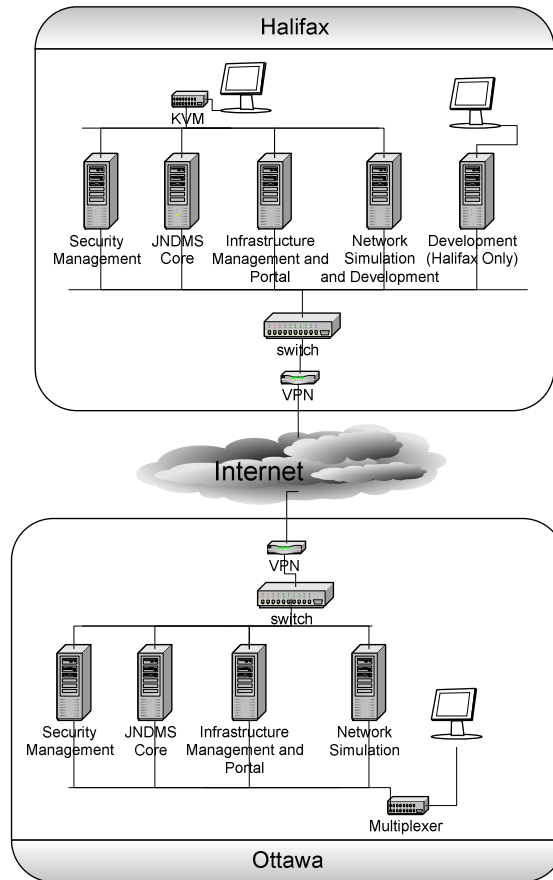


Figure 2-2: JNDMS Physical Environment

Table 3 provides a brief description of the physical workstations to be used, as well as a note on the use of virtual machines. The use of virtual machines for the development serves multiple purposes. Some of the virtual machines are used to simulate additional hosts to ease development and testing, while others are used to allow a geographically diverse development team easier management of the system. The IP field is unique to each configured environment, either physical or virtual.

Table 3: Cycle 3 Workstation Configuration

Computer name	Location	IP	OS	CPU	RAM	HD	Model	Primary Roles
HALIFAX								
Overseer	Halifax	142.128.80.140	RH ES 4	1x 2.8 Ghz	4 GB (1.5 GB)	1 x 250	SC1420	DNS
jndms_svn	vm - Overseer	142.128.80.142	Fedora C4		1.2 GB			SVN, Xwiki
Gatemaster	Halifax	142.128.80.150	RH ES 4	1x 2.8 Ghz	4 GB	2 x 250	SC1420	Oracle DB, ISM
caump01	vm -Gatemaster	142.128.80.151	Win 2003		1 GB			Host SIM
Protector	Halifax	142.128.80.160	Win 2003	1x 2.8 Ghz	2 GB	1 x 250	SC1420	ArcGIS,Portal
WatchDog	Halifax	142.128.80.180	Fedora C4	1x 2.8 Ghz	4 GB (2 GB)	1 x 250	SC1420	VM Server
cauni01	vm - Watchdog	142.128.80.181	Win 2003		1.3 GB			Unicenter
Breakdown	Halifax	142.128.80.170	Fedora C4	1x 2.8 Ghz	4 GB (2 GB)	1 x 250	SC1420	VM Server
Unclas	vm-Breakdown	142.128.80.171	Win XP		512 MB			Unclas system
Shield	Halifax	142.128.80.190	Win 2003	2x DC 3.2 Ghz	4 GB	2 x 250	1900	RouterSIM/Hudson
Spectrum	vm - Shield	142.128.80.191	Win 2003		1.5 GB			Spectrum
NRNS								

DN0678
Issue 4/2: 06 February 2012

Computer name	Location	IP	OS	CPU	RAM	HD	Model	Primary Roles
Champion	NRNS	192.168.200.2	RH ES 4	1x 2.8 Ghz	4 GB (1.0 GB)	1 x 250	SC1420	DNS
cauni01	vm - Champion	192.168.200.111	Win 2003		1 GB			Unicenter
caump01	vm - Champion	192.168.200.112	Win 2003		1 GB			Admin
Spectrum	vm - Champion	182.168.200.118	Win 2003		1 GB			Spectrum
WatchOfficer	NRNS	192.168.200.3	RH ES 4	1x 2.8 Ghz	4 GB	2 x 250	SC1420	Oracle DB
ISM_551	vm - WatchOfficer	192.168.200.142	Win 2003		?			ISM
ISM-Client	vm - WatchOfficer	192.168.200.141	Win XP		256 MB			ISM-Client
Sentinel	NRNS	192.168.200.4	Win 2003	1x 2.8 Ghz	2 GB (1.0 GB)	1 x 250	SC1420	ArcGIS,Portal
JSS-DSS	vm - Sentinel	192.168.200.120	Win 2003		512 MB			JSS-DSS
Aion-IDE	vm - Sentinel	192.168.200.121	Win XP		512 MB			Aion-IDE

2.5 System Security Overview

This section outlines security guidelines for some of the JNDMS interfaces.

The user roles will be managed by the JNDMS core services. The Data Warehouse will be configured so that it will communicate with other JNDMS components and not directly with the user so that the user's profile will be managed by the core services. Once the user has been authenticated there will be no internal segregation of data for each user. Each profile has access to the SA.

The user interface component communicates with the system through web standards. The configuration will use basic authentication although stronger security, such as the use of SSL, can be configured through the portal.

The communications between JNDMS remote components is primarily through Web Services. Basic authentication over HTTP can be used. The standards and products for web services offer enhanced security measures that can be enabled.

There are a number of inputs to JNDMS that are managed through external tools, such as NSM and Intellitactics. The security between the sensors and these tools are managed externally to JNDMS.

2.6 Model Relationships

The relationships identified here should provide insight into the fundamental reasons for capturing the data. This section identifies primary entities, secondary entities and the relationships between the primary entities.

2.6.1 Primary Entities

The primary entities represent the fundamental concepts that must be displayed and managed within JNDMS. These are the items that must have all of the details captured and are the items that can be explored directly as part of the visualization.

- Assets
- Vulnerability Definitions
- Safeguards
- Operations
- Locations
- Zone
- Events/Incidents

2.6.2 Secondary Entities

The secondary entities may involve important concepts but will often be represented as attributes of the primary entities or included in the information on links, relationships or in the details of the other entities.

- Exploits
- Signatures
- Malware
- Products, vendors
- Operational Units
- Operational Events
- Zonesubnet
- Site (relates one or more locations)
- Scans
- History records
- Risk (generally not a separate entity but an attribute of many of the primary entities that must be shown)
- Points of Contact
- Requests for Change (RFC)

2.6.3 Entity Relationships

The following tables show the relationships between the primary entities. Some of the shown relationships actually traverse a few links to get to their target entity which in some cases may be redundant. For example a location links to safeguards through assets, but we could also follow the links directly from location then to the safeguard. This was done intentionally to show the most immediate information and to provide the entire relevant context. The direct links are the most important and the usefulness of the other links may have to be investigated.

2.6.3.1 Link Summary

Source	Link	Link	Description
Any	Direct	Any	Direct links.
Asset	Any	Asset	The type of link, and therefore the colour of the link should depend on the type attribute of the endpoints. The specific relationships are shown in the 'asset links' section below.
Asset	Vulnerability instance	Vulnerability Definition	The vulnerability instance shows that this particular asset is vulnerable.
Asset	Safeguard Impl.	Safeguard	
Asset	OpAsset / OpDetail	Operation	This link creates a relationship between operations and the assets the operations depends on (this relationship includes attributes to qualify the importance of the asset to the operation).
Asset	Op asset / op area	Location	This link shows the relationship, but full location information is found in location table.
Asset	Zone border	Zone	Assets that create zones.
Asset	Zone subnet	Zone	Assets in a zone.
Asset	Sensor (attrib)	Incident	Sensor reporting incident.
Asset	affectedAsset (attrib)	Incident	Impact on asset.
Asset	sourceIP	Incident	Source of attack.
Asset	Vuln. instance	Incident	Vulnerability instances as a result of an incident.
Vulnerability Definition	Vuln. Inst.	Assets	Port information is important for this link. Colour of vuln inst links should also show if there is currently a safeguard in place for this vulnerability instance.
Vuln def		Vuln def	No current relationship but under review for IAT.

DN0678
Issue 4/2: 06 February 2012

Source	Link	Link	Description
Vuln def	Vuln inst / safeguard protection	Safeguards	
Vuln def	(risk)	Operation	No direct link but could show how this vulnerability relates to the risk in the given operation. The link could show risk.
Vuln def	Vuln. Id	Incidents	If the vuln id and service id are valid the link would refer to a vuln. Instance.
Safeguard	Safeguard impl.	asset	Asset providing safeguard.
Safeguard	Safeguard prot/vuln instance	Asset	Asset being protected. Colour should indicate effectiveness of safeguard.
Safeguard	Zone border	Asset	Zone border asset.
Safeguard	Safeguard prot/vuln instance	Vuln def	
Safeguard	Zone border	Zone	Creation of zone.
Safeguard	Safeguard impl / zone subnet	Zone	Safeguard in zone.
Safeguard	Safeguard prot / zone subnet	Zone	Safeguard protecting in zone.
Operation	Opasset	Asset	Assets needed by operation. Link attributes/colour could indicate risk associated with asset.
Operation	(risk)	Vuln def	Show risk for this operation represented by vulnerability.
Operation	Op asset / safeguard impl	Safeguards	Safeguards protecting operation.
Operation	Op area	Locations	
Operation	Op asset / zone subnet	Zone	Zones required by operation.
Operation	Op asset / sensor attrib	Incidents	Activity on sensors used by this operation.
Operation	Op asset / affected attrib	Incidents	Incidents impacting this operation. Link could indicate activity (count of incidents) or risk.
Locations	Op area	asset	Assets at location. Link could show associated risk.
Locations	Op area / asset / vi	Vuln. Def	Vulnerabilities at location. Link could show associated risk.

DN0678
Issue 4/2: 06 February 2012

Source	Link	Link	Description
Locations	Op area / asset /Safeguard impl	Safeguards	Safeguards deployed at this location.
Locations	Op area / asset / zone subnets	Zones	Zones at location.
Locations	Op area / asset / sensor att	Incidents	Active sensors at location.
Locations	Op area / asset / affected attrib.	Incidents	Assets impacted at location.
Zone	Zone border	Asset	Created by
Zone	Zone subnet	Asset	Includes
Zone	Zone border / safeguard impl	Vuln def	Vulnerabilities blocked by zone border (could reference rules).
Zone	ZoneBorderSafeguard	Safeguards	Protection provided by zone.
Zone	Zone subnet / asset	Operation	Operations affected by zone.
Zone	Zone subnet / asset	Location	Locations affected by zone.
Zone	ZoneLocations	Location	Default locations for zone. The actual locations are defined by the locations of the assets within the zone (see above).
Zone	Zone border	Zone	This would show communications between zones. A useful piece of information for this link would be if there are any know threat vectors.
Zone	Zone subnet / asset / sensor attrib.	Incidents	Active sensors within zone.
Zone	Zone subnet / asset / affected	Incidents	Incidents impacting zone.
Events/Incidents	parentIncidentID (attrib)	Events/Incidents	This represents a cause and effect type relationship between incidents or events.
Events/Incidents	IncidentCorrelation	Events/Incidents	This represents a correlation or relationship
Events/Incidents	Sensor (attrib)	Asset	Asset that detected the sensor.
Events/Incidents	affectedAsset (attrib)	Asset	Asset that is impacted by incident.
Events/Incidents	sourceIP	Asset	Source of incident.
Events/Incidents	sensor / asset / op area	Location	Location of sensor.

DN0678
Issue 4/2: 06 February 2012

Source	Link	Link	Description
Events/Incidents	Affected / asset / op area	Location	Location of affected asset.
Events/Incidents	Sensor / op asset	Operation	Activity of sensors used by operation.
Events/Incidents	Affected / op asset	Operations	Direct impact on operations. This can also examine the impact fields.
Events/Incidents	Impact / risk	Operations	Impact to operational risk by this event.
Events/Incidents	Vuln. Id	Vulnerability Instances	Vulnerabilities referenced as part of this event.
Events/Incidents	Sensor / asset /zone	Zone	Zone where sensor resides. The can be extended to view incidents/attacks that span zones.
Events/Incidents	Affected / asset /zone	Zone	Zone where affected asset resides.
Events/Incidents	Sensor / safeguard impl	Safeguards	Identify if this sensor is a safeguard.
Events/Incidents	Affected / safeguard impl	Safeguards	Identify if the affected asset is a safeguard.
Events/Incidents	Vuln id / safeguard	Safeguards	Identify safeguards against this vulnerability.

2.6.4 Asset links

This section discusses how JNDMS infers the relationships between assets using its data model.

The JNDMS data model does not directly attribute a link but depends on inferring the meaning of the link from the endpoints of that link. The dependency links are built from the dependent table (AND relationships) and the redundant table (OR relationships). (See sections 3.5.4.3.1 and 3.7.7.1.2 for details.)

Figure 2-3 shows some examples of working out the logical connections between assets with some discussion afterwards of the constraints we can place on this model and a list of the possible asset categories.

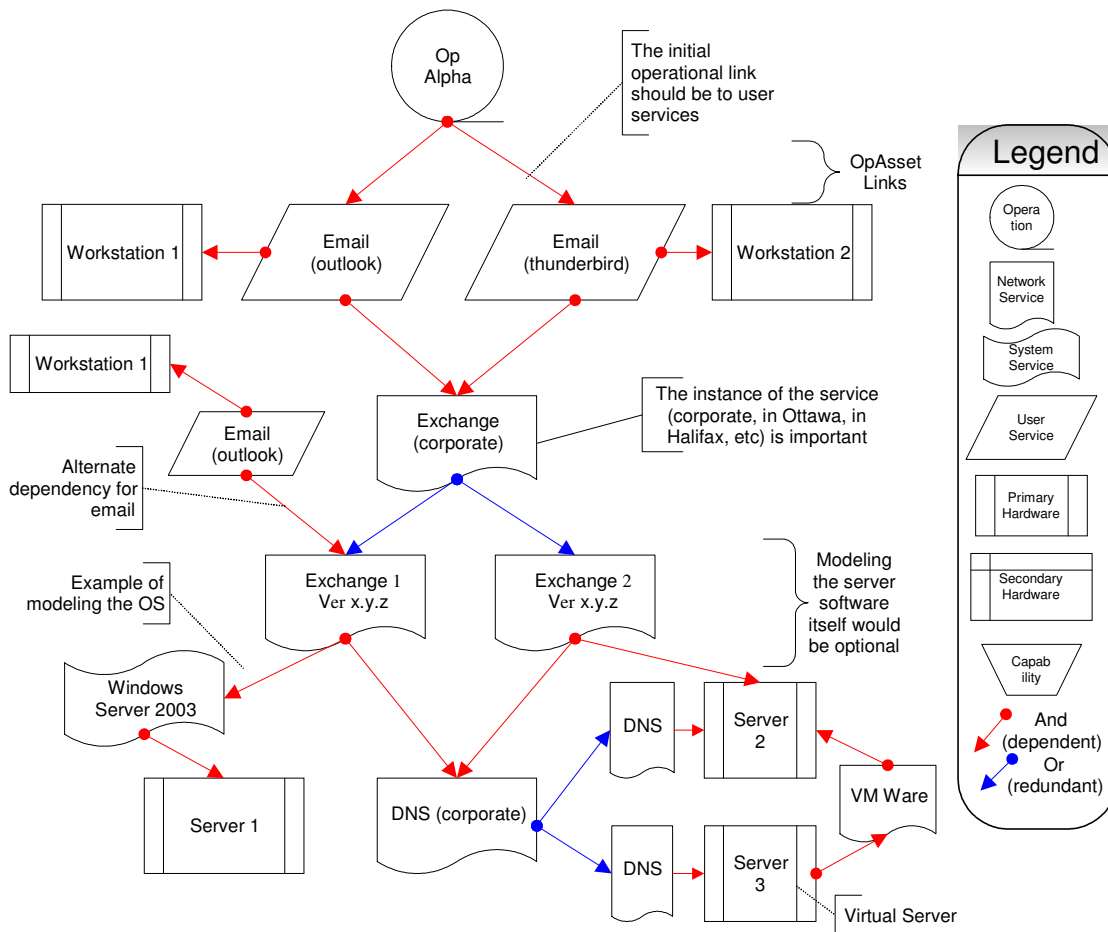


Figure 2-3: Connections Between Assets

The example (Figure 2-3) illustrates that an operation (*Op Alpha*) depends on two user services (OpAsset links): *Email (Outlook)* which in turns depends on *Workstation 1* (a physical host), and *Email (thunderbird)* with in turn depends on *Workstation 2* (a physical host). Because these two email services are required and they run on different machines, JNDMS understands that both workstations are dependencies of *Op Alpha*.

Email network services are provided by a 'logical service' identified as *Exchange (corporate)*. The actual network service is provided by two redundant instances of Exchange Server. This is represented by the 'or' arrows between *Exchange (corporate)* and the two *Exchange (Ver x.y.z)*¹ instances. The *Exchange 1* instance is shown to depend on a system service (*Windows Server 2003*) hosted on the physical hardware (*Server 1*). While the *Exchange 2* instance is hosted on *Server 2* (in this example the operating system for *Server 2* has not been modeled).

Both instances of Exchange server depend on the corporate DNS service (a logical network service). This service is provided by redundant DNS servers hosted on *Server 2* and *Server 3*. In this example, *Server 3* is a virtual server running on (dependent on) *VM Ware* which is hosted on *Server 2*.

¹ Ver x.y.z is to indicate that these are the actual instances of the Exchange server (i.e., specific versions) in contrast to the 'logical' instance identified as the corporate service.

Asset Type Categories

The JNDMS system categorizes assets into the following 6 types. Each type is subdivided into a number of categories. The JNDMS Data Warehouse stores the available asset types and categories in the ASSETCATEGORY table.

- **User Service.** Service entry points or software specifically identified as a user requirement. For instance, email services provided via an email client such as Outlook, or Web-based service accessed through a browser such as Internet Explorer, etc.
- **System Service.** Locally installed software or services such as the operating system, background services/daemons or other software not directly identified as a user requirement. By default, software not specifically identified in other categories (i.e., user service) will be identified as a system service when reported by the asset management discovery tool. The development dataset stores all installed software, including user-related software, as System Services.
- **Network Service.** A service that is provided across the network. Enterprise email servers (i.e., MS Exchange), Web servers, DNS servers are examples of network services.
- **Primary Hardware (includes virtual hardware).** This would represent the key hardware that must be tracked and understood by JNDMS. This would include workstations and servers (host types), routers and switches (network infrastructure) and other hardware with locations of their own. The primary hardware entries can stand on their own and do not require a relationship to other hardware entities.
- **Secondary hardware.** This type of hardware must have a relationship to primary hardware through the 'network host' link (in the Asset table). This type of hardware would include components or peripherals that are tracked and monitored for the purposes of JNDMS. The network interfaces come under this category because they are part of a host (or router, or similar), are required to fully understand the connectivity (including multiple IP addresses) but generally will not be considered directly by JNDMS rule processing.
- **Capabilities.** This would represent one or more user services. For example a report may require access to email and a particular web portal.

Table 4 shows the count of all assets by type and category in the development dataset.

Table 4: Asset Count by Type and Category in Development Dataset

COUNT	TYPE	CATEGORY
6	network service	service
13	primary hardware	firewall
5246	primary hardware	host
275	primary hardware	network switch
38	primary hardware	router
5648	secondary hardware	network interface
205107	system service	software

Rules and Guidelines for links

There are a number of rules and guidelines to keep in mind when linking operations to assets, or creating asset dependencies and redundancies links. These rules may be enforced by the JNDMS core components and used when interpreting the results. The following rules and guidelines are used:

- Operations dependencies on Assets (OpAssets) will link Operations to capabilities, user services, network services or primary hardware.
- An OpAsset link can be identified as access only or a provision (opasset.provision = 'Y' or 'N'). Access only means that there is only a need to access the asset, it will remain under the control of another operation. Provision means that the asset will be provided and/or controlled by the operation itself.
- When an OpAsset link is assigned as “provision”, the dependency tree will be automatically searched to find the primary hardware. It will have an OpAsset record added and flagged as implied (opasset.implied = 'Y').
- User services have at least one dependency on hardware to identify where it is installed or accessed from. This can be a direct link to the hardware or through one or more other system services (locally installed software).
 - The user service may also have one or more dependencies on a network service to identify a client/server relationship.
 - Links to other user services would be unusual but may indicate dependences within the host.
- Network services. A network service represents a service that requires a communications link to access. This may be a purely logical service such as email, or the specific software installed that provides the services, such as Exchange. Figure 2-3 (above) shows both.
 - Any link to hardware indicates where it is installed. A logical service may have no links to hardware if it is purely logical. This would be shown as having only relationships to other network services. If a network service is installed software then it must have links to hardware either directly or through one or more system services.
 - Links to other network services identify dependant/redundant services.
 - Links to system services show relationships within a host, such as modeling the dependence of a service on the operating system. The link to the hardware this service is installed on may be through system services.
 - A network service must either link to one or more network services OR link to hardware. The link to hardware can be a direct link or indirect through system services.
- Capabilities must reference one or more user services.
- Hardware nodes are the only entities that represent actual hosts. All communications would be addressed through these hardware nodes. In the

case where virtual hardware is represented it would still show up as a hardware node, although it may have dependencies on other nodes.

Network Links

- Physical links, through the links table, must go through a network interface. No links will join primary entities directly. There is no expectation that network interfaces have an IP address to allow for connections that may not be OSI layer 3. This would also require any primary hardware entity that has a connection to have at least one network interface, even if a 'default' interface is created for model consistency.
- VPNs and other tunneling technology such as crypto gear are modeled using the network links as well. The physical connection should be modeled for JNDMS, however the VPN router should be identified as a zone border with rules that identify the following:
 - Traffic is only allowed to flow to and from target zones.
 - The traffic undergoes a transformation (encrypted at one end, decrypted at the other).

A summary of asset types and the relationships of their links:

Source Type	Dest. Type	Link Table	Required	Description
User Service	Hardware	Dep	The link to primary hardware must be available but can be direct or through one or more system services.	Software installed on a particular asset.
User Service	System Service	Dep/Red		Shows relationships between components on a host.
User Service	Network Service	Dep/Red		Shows a relationship that requires communications such as client/server.
User Service	User Service	Dep/Red		This is a little unusual but may appear if system services are also directly required by the user. Treat this as a link to a system service (local).

DN0678
Issue 4/2: 06 February 2012

Source Type	Dest. Type	Link Table	Required	Description
System Service	Hardware	Dep/Red	The link to hardware is required but can be direct or through other system services.	Installed on. Either software modeled or not.
System Service	System Service	Dep/Red		Relationships on a host.
System Service	User Service	Dep/Red		Same as the system service to system service relationship. Some system services may also be identified as specific user needs.
System Service	Network Service	Dep/Red		Shows a relationship that requires a communications link such as client/server.
Network Service	Hardware	Dep/Red	A network services must have either a link to primary hardware (possibly through one or more system services) OR links to other network services (purely logical service)	Installed on.
Network Service	System Service	Dep/Red		Shows relationships within a host on how the network service is implemented.
Network Service	User Service	Dep/Red		Shows relationships within a host (same as links to system services).
Network Service	Network Service	Dep/Red		Shows relationships between network services.
Capability	User Service	Dep/Red	One or more user services required	A capability is the combination of one or more services.
Primary Hardware	Primary Hardware	Dep/Red		May show a virtual hardware relationship.

DN0678
Issue 4/2: 06 February 2012

Source Type	Dest. Type	Link Table	Required	Description
Secondary Hardware	Primary Hardware	Network Host (in the Asset table)	Every secondary hardware must have a link to the primary hardware through its 'network host' link	Shows a relationship of components or peripherals.
Secondary Hardware	System Service	Link	Links are through the network interfaces of the host or primary entity	Communications link.

3 Subsystem Design

This section describes the details of each of the JNDMS subsystems. These subsystems are shown in Figure 3-1.

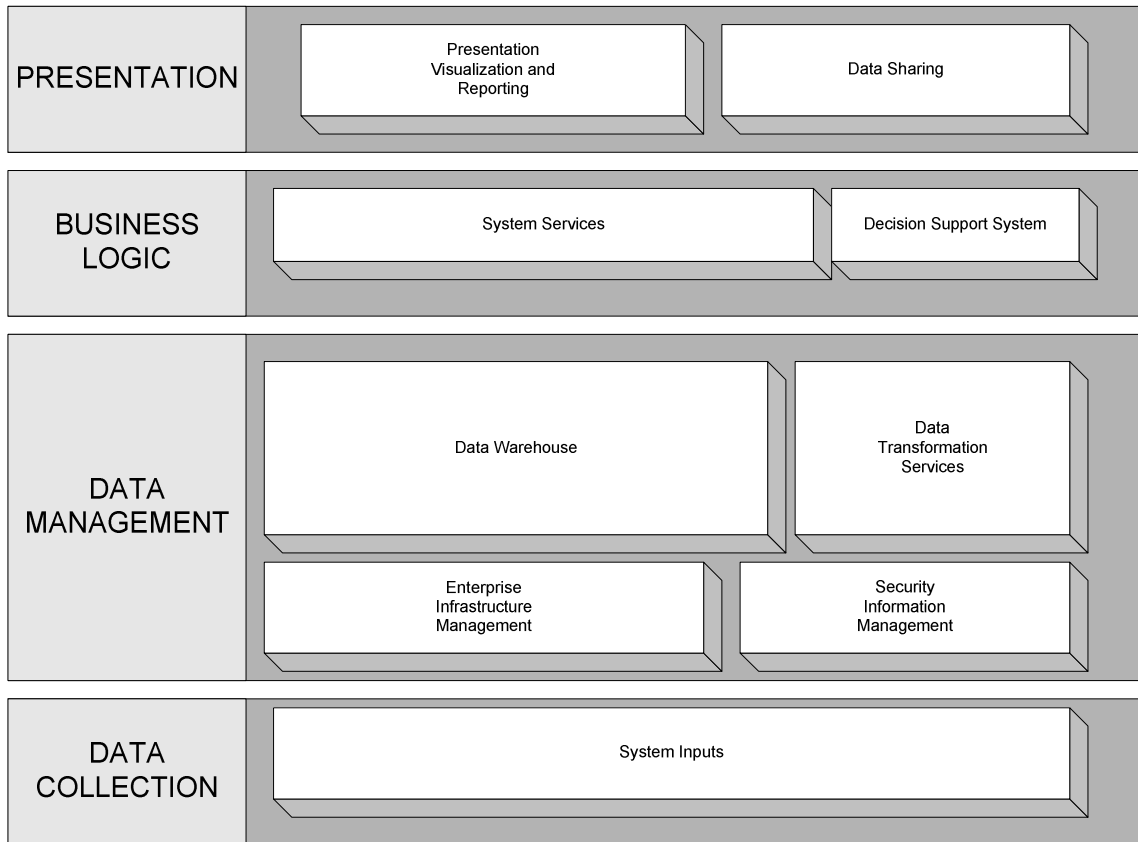


Figure 3-1: JNDMS Subsystems

3.1 System Inputs

Situational Awareness for Computer Network Defence will be achieved by combining the data from the following five domains: IT infrastructure and services, military operations, vulnerabilities and exploits, safeguards and security events.

An organisation should have all the tools in place to collect this information. In reality, those tools are often incomplete and not integrated with each other. JNDMS' main goal is not to deliver system input tools, however, this TD Project (TDP) will ensure that the right information is fed into JNDMS, and will be able to make recommendations for the future deployment of JNDMS. The tools selected to front end JNDMS are discussed in the appropriate "COTS Selection" section under each domain (sections 3.1.2 to 3.1.6). Section 3.1.1 provides an overview of the terminology used in JNDMS.

3.1.1 Terminology

3.1.1.1 Incidents

The following terms are used to describe incidents in JNDMS (in sequence of severity): Event and Security Incident.

An Event is an observable change of state in the CND environment.

A Security Incident is an event (or group of events) that impacts the Confidentiality, Integrity and/or Availability of an IT asset.

These definitions are useful to make an initial distinction between the sources of incidents within JNDMS. The EIM and SIM subsystems detect security incidents. The DSS subsystem correlates and contextualizes all incidents, including the tickets from the organization's ticketing tool. The ticketing tool, discussed in section 3.1.6.1.2, reports both types of incidents.

3.1.1.2 IT Infrastructure: Static versus Dynamic

The RFP defines the static IT infrastructure as the "manually updated configuration databases, resulting from policies and IT management and implementation processes". The configuration databases capture the baseline IT infrastructure data that JNDMS must work with.

The dynamic IT infrastructure is the data captured during automatic discovery. It captures the current perspective of the IT infrastructure.

In many cases JNDMS does not explicitly differentiate between these two types of events. The data sources reporting to, or queries by, JNDMS may be of either type but the resulting event generation and resulting analysis may not be impacted.

3.1.1.3 Vulnerability: Definition versus Instance

A vulnerability is a condition that could allow assets to be harmed by an attack.

A vulnerability definition is the knowledge of a flaw in a type of system or software. There are thousands of vulnerability definitions captured in online vulnerability databases. Those databases attempt to assemble all publicly known cyber vulnerabilities.

A vulnerability instance is a vulnerable asset within the organisation. It can only be discovered when the IT infrastructure is assessed against known vulnerability definitions. A vulnerability instance is the link between vulnerability definitions and the organisation's IT infrastructure.

3.1.1.4 Safeguards: Mitigation versus Safeguard

A safeguard is a security measure that prevents or reduces the risk of exploitation of specific vulnerabilities. It can completely reduce the risk (resolve it).

A mitigation is the implementation of a safeguard to reduce the risk.

3.1.1.5 Exploit versus Security Incidents

An exploit is a method that could be used to compromise Confidentiality, Integrity and/or Availability (C, I or A) through one or more vulnerabilities. Exploits are captured similarly to the JNDMS vulnerability definitions and they should be linked to one or more vulnerability definitions. An exploit can be deliberate or accidental, physical or software related. This definition is consistent with the common use of exploit as a noun referring to a specific piece of software or methods that can be used to exploit vulnerabilities. The actualisation of an exploit is a security incident.

A threat agent is an entity that may act to cause a security incident. The threat agent can be human or non-human. Human threats can be internal (people who have some level of authorized access to the IT systems) or external (people who don't). Non-human threats can be deliberate and initiated by humans (malware, application program exploits, explosive devices, jammers, fire) or accidental (IT malfunctions, Acts of God, physical environment [electrical, fire, water and air related]).

3.1.1.6 Knowledge Base versus Event

In this design, the knowledge base is considered a time invariant description of the computer network environment, the IT services value and the potential threats (IT infrastructure, safeguards, military operation requirements, and vulnerability and exploit definitions).

Contrary to the knowledge base, events are dynamic and must be sensed. Similar to incidents, which are detected through sensors, vulnerable assets are unknown until the environment is scanned (or assessed). Incidents cause actual damage and vulnerability instances represent potential damage. These events are sensed and contextualized by JNDMS. The DSS will identify incidents and vulnerability instances. This task is supported by a front end COTS, including the EIM subsystem (section 3.2), the SIM subsystem (section 3.3) and Vulnerability Scanners (section 3.1.4.1). Incidents and vulnerability instances are discussed in the DSS subsystem section (section 3.7.1).

3.1.2 IT Infrastructure Data

The infrastructure is essential to the delivery of IT services to the organisation. The IT infrastructure includes all IT equipment on the network, their applications, the services provided and all interrelationships. In addition, the physical location of this equipment must be captured, as well as the telecommunication circuit data.

In order to assess the IT infrastructure readiness, active monitoring of the infrastructure must be achieved to show both the actual layout of the IT infrastructure and its potential or current exposures.

The physical locations of the equipment will be entered manually into JNDMS. These locations are necessary to display the IT infrastructure geospatially. The connectivity data is available from the National Telecommunication Management System (NTSM). It will be exported or entered manually.

DND has approximately 400 Service Delivery Areas (SDA) nationally. An SDA is typically a base, a station, a reserve unit or a National Defence Headquarters (NDHQ) building. A SDA is comprised of one or more Service Delivery Points (SDP). A SDP is a precise location, typically a piece of equipment located in a room or other facility where DND users connect to the service cloud; for example, Private Branch Exchanges (PBX), multiplexers (MUX), routers, Main Distribution Frames (MDF), etc. There are approximately 1300 SDPs nationally. A SDP contains one or more Service Interface Points (SIP). A SIP is the physical interface at a SDP. Each SIP is defined as the combined physical, electrical and service interface and its key attributes (Class of Service [CoS], Quality of Service [QoS], Mean Time To Restore Service [MTTRS], physical interface, protocol, transport technology, circuit numbers, access data rate, user and application specific information).

3.1.2.1 COTS Selection

Different products can be used to manage the IT infrastructure network devices, workstations and servers' data inputs. They are:

The Unicenter Network and Systems Management product line.

Unicenter NSM can provide both real-time and historical performance monitoring to both types of IT infrastructure devices (network and servers).

(A locally deployed agent will be required to get detailed monitoring of infrastructure servers.)

The Concord product line.

Concord products (eHealth and Spectrum) can provide both real-time and historical performance monitoring to both types of IT infrastructure devices (network and servers).

(A locally deployed agent can be used to get detailed monitoring of infrastructure servers.)

Unicenter Asset Management

Unicenter Asset Management (AM) provides servers' detailed hardware and software inventories.

(A locally deployed agent will be required to get the detailed servers inventories.)

CentennialDiscovery

Centennial is an alternate source for software inventory. This is the COTS that was deployed as part of the final DREnet deployment.

All of these products are capable of providing some form of static and dynamic data inputs. For network devices, it is assumed that all relevant information will be available through Simple Network Management Protocol (SNMP) agents, running locally on the network devices themselves.

It should be noted that the commercial products that target IT infrastructure management are continually improving. The goal of the JNDMS TD is to assess the best products to gather the required information when that information must be integrated into the system.

During the development of JNDMS it became apparent that the base discovery information of the network topology was essential to inferring much of the impact of various events. It was found that different network management systems will report on the topology in different ways. Some tools report an approximate logical representation, while others report on the details of the physical connections. Spectrum was chosen for the core discovery information because of its ability to model the physical topology, at the connection level.

3.1.3 Military Operations Data

The military operations data is essential to assess the value of each IT infrastructure asset. The organisation depends on the services provided by the IT infrastructure. The military operations data should capture the IT services requirements for all domestic, international or deployed operations. JNDMS will provide an interface to capture the importance of any remaining IT services not captured by a military requirements tool.

The requirements should capture the operation priority and the operation events. An operation event may be a location change for example. Each operation event has a priority and IT service requirements. The priority of an event may be equivalent to the operation priority, or less. The IT service requirements include the service importance and the service values in terms of Confidentiality, Integrity and Availability.

3.1.3.1 Sources of Operations Data

There is no current active data source that can meet the requirements of the JNDMS in describing the operation and its IT Infrastructure requirements. The Command and Control Information Exchange Data Model (C2IEDM)/JC3IEDM provide a standard for operational data exchange but there is no current source identified. Perhaps, the Operations Database (ODB) can be examined in the future for the purpose of operational data exchange.

For the purposes of the JNDMS TD test, trials and simulations, it is anticipated that the initial operations data will be extracted from IAT. Some values for the JNDMS Military Operations data set will be generalized from the load process. For a detailed description of the Military Operations Data see section 3.5.4.2.2 (Military Operations Data).

Table 5 shows the operational information available from IAT:

Table 5: Operational Attributes from IAT

General Information	Details	Values
Operation Name		(string)
Operation Details	Group	A6
		G6
		J6
		N6
	Type	International
		Deployed
		Domestic

DN0678
Issue 4/2: 06 February 2012

General Information	Details	Values
	Strength	(number)
	Operation Priority	Critical
		High
		Medium
		Routine
		Exercise
	N/A	
Start Date		
Expected End Date		
Operation Locations	Location	(list of bases and deployed locations)
	Location Priority	Critical
		High
		Medium
		Low
		N/A
Operation Log	Dates	
	Notes	
	User	
IT Services	Location	(pick from locations above)
	Connection Type	MT Heavy Det
		InMarsat / Light det
		Iridium
		SGT Adco
		VSAT
		HSD
		SAC
T1 + (TSRP)		

General Information	Details	Values
		DISAP
		Local telco (dial-up)
		Local telco (high speed)
		HF Data / Voice (QRT)
	ID	
	Network	(pick from the list of networks)
	Bandwidth (Kbps)	
	IT Service	(pick from a list of IT services)
	IT Service Type	Admin / all purpose
		Command and Control
		Morale / other
		Operational / all purpose
		Voice
	Value	Essential
		Very important
		Important
		Useful
		N/A

3.1.4 Vulnerability and Exploit Data

The JNDMS vulnerability database should integrate all publicly known cyber vulnerabilities at the disposition of the organisation and physical vulnerabilities. Due to the nature of vulnerabilities, there are multiple public vulnerability sources (or resources) from the industry to government agencies. The Internet has, thus, become the preferred media for sharing and publicizing vulnerabilities.

It is expected that the content of non-public source of vulnerabilities, such as the intelligence coming from the Allies, will not be integrated into JNDMS unless the instance resides on the appropriate classification level. Non-public exploit data is typically classified.

The vulnerability database will track the following:

1. The vulnerability definitions and descriptions
2. The exploit information related to each vulnerability definition
3. The solutions (or safeguards) to resolve the vulnerability or a specific exploit (patch or other fix)

The information in this database is used to perform risk assessment of known vulnerability instances; it can also be used to assess the IT infrastructure. COTS vulnerability scanners will also be used to support vulnerability assessment.

3.1.4.1 COTS Selection

There are multiple online sources of vulnerabilities. JNDMS will be able to benefit from one important achievement:

- Common Vulnerability Exposures (CVE) established a common vulnerability naming and enumerating standard. The United States Department of Homeland Security founded CVE and it is now maintained by the MITRE Corporation. The naming convention is currently well adopted. CVE is a dictionary, not a database: it mostly lists vulnerabilities and their references. There are currently approximately 150 CVE-compatible products or services. The list includes CA, CERT, CISCO, Nessus, National Vulnerability Database (NVD), Open Source Vulnerability Database (OSVDB) and Snort.
 - CA eTrust Vulnerability Manager is a vulnerability management tool and CA eTrust Policy Compliance is a configuration management, risk assessment, and policy compliance tool. CA's vulnerability data contains CVE references that are constantly updated and are based on the most recent updates made by MITRE.
 - The CERT Coordination Centre uses CVE and contributes to new CVE entries.
 - CISCO Secure Intrusion Detection System incorporates the CVE dictionary.
 - The Nessus Security Scanner is a vulnerability assessment tool. Nessus reports now include relevant CVE names.
 - The NVD from the National Institute of Standards and Technology (NIST) is an online vulnerability database. The database contains all CVE information, as well as vulnerability attribute information.
 - Snort is an open source network intrusion detection and prevention system. The CVE reference is included in the signature description.
 - OSVDB is an independent and open source database created by and for the community. The database contains a full mapping to CVE entries.
 - NCircle IP360 is a vulnerability scanner that can report on CVE.
 - Centennial Discovery has a component that can report on likely vulnerabilities, and can provide CVE references.

3.1.4.2 Vulnerability Database

The integration of all known vulnerabilities and exploits is a project in itself. The Communications Security Establishment (CSE) is in the process of getting a similar database. Until JNDMS can benefit from this undertaking, we propose to use the most complete database available.

3.1.4.2.1 The Online National Vulnerability Database

NVD is a comprehensive cyber security vulnerability database that integrates all publicly available United States Government vulnerability resources. The vulnerabilities have also been analyzed, which will be useful input to JNDMS. NVD provides CVSS attributes of the base metric. CVSS is discussed in section 3.7.2.6 (Severity Assessment). The CVSS attributes allows for a greater variety of risk calculations to be performed on vulnerability instances automatically (without requiring analysis by a human). Perhaps most important is being able to differentiate those exploits which simply require remote access rather than local access.

Table 6 outlines information available from the NVD:

Table 6: Vulnerability Information from the NVD

Vulnerability Information	Value
CVE ID	CVE_year_####
Vulnerability Description	String
Vulnerable Software and Versions	(normalized product names and versions)
Released Date	
Access Vector (CVSS Base Metric)	<ul style="list-style-type: none"> • Remote • Local
Access Complexity (CVSS Base Metric)	<ul style="list-style-type: none"> • High • Low
Authentication (CVSS Base Metric)	<ul style="list-style-type: none"> • Required • Not Required
Confidentiality Impact (CVSS Base Metric)	<ul style="list-style-type: none"> • None • Partial • Complete
Integrity Impact (CVSS Base Metric)	<ul style="list-style-type: none"> • None • Partial • Complete
Availability Impact (CVSS Base Metric)	<ul style="list-style-type: none"> • None • Partial • Complete
Authentication Impact	<ul style="list-style-type: none"> • Administrative privileges (0 or 1)

Vulnerability Information	Value
	<ul style="list-style-type: none"> • User privileges (0 or 1) • Other privileges (dependent on program being exploited) (0 or 1)
Vulnerability Type	<ul style="list-style-type: none"> • Access validation error (0 or 1) • Input validation error (0 or 1) • Design error (0 or 1) • Exception condition error (0 or 1) • Environmental error (0 or 1) • Configuration error (0 or 1) • Race condition error (0 or 1) • Other error (0 or 1)
Advisory Information	Text description
Patch Information	Text description

Another important feature of the NVD is that the CVE entries are available in Extensible Markup Language (XML) format, along with the CVSS base metric attributes and a list of normalized product names and versions affected for each year starting in 2002. For the current year, a Really Simple Syndication (RSS) feed is available to publish new vulnerabilities to subscribers, or a versioning timestamp can be used to calculate deltas. The total size of all NVD data from 2002 to present is approximately 20 MB.

The normalized list of product names and versions allows for vulnerability instances to be projected from a configuration management database of assets using the same normalized product list.

3.1.4.2.2 Impact Assessment Tool

The Canadian Forces Network Operation Centre (CFNOC) Network Vulnerability Assessment Team (NVAT) currently uses IAT to log vulnerabilities. Table 7 outlines the Vulnerability and Exploit data available in IAT.

Table 7: Vulnerability and Exploit Data Available in IAT

Vulnerability	Vulnerability #	
	DND Adv. #	
	CVE #	
	Vulnerability Name	(string)
	Opened Date	
	Released Date	
	Vulnerability Type	Design / embedded flaw
		Implementation / system architecture
		Configuration / settings
		Policy / usage
		Unknown
	Status	Active
		Mitigated
		Forecast
		Resolved
	Description	(string)
Report Confidence (CVSS temporal metric)	Confirmed	
	Unconfirmed	
	Uncorroborated	
Exploits	Name	(string)
	Availability (CVSS temporal metric)	Public
		Within Int community
		None
	Public Date	
	Main Source (CVSS base metric)	Remotely (internet)
		Internal (intranet)
		Local (host)
Unknown		
Access Type	Authentication	

	(CVSS base metric)	Anonymous	
	Maturity	proof of concept code	
		exploit code	
		tool integration	
		worm integration	
	Damages (CVSS base metric)	Loss of data confidentiality	
		Disruption of data or system integrity	
		Disruption or Denial of availability	
		Loss of reputation	
		Financial loss	
		Injury-life	
	Damage bias (CVSS base metric)	100%	
		67%	
		50%	
		33%	
		25%	
		0%	
	Affected Assets (vulnerability instances)	Asset type	Equipment
			System service
User service			
Asset Name & Vendor		(pick from a list)	
Asset Model-Detail		(string set in Asset Name list)	
Network and designation		(list of networks)	
Description		string	
Mitigation	Mitigation Types	Guards	
		Gateways	

		Authentication devices
		Backups
		Cryptographic checksum
		Cryptography
		Secure links
		Software patches/upgrades
		Tunnel (IPSec, SSL, ..)
		Virus scanners
		VPN
		Firewall
		IDS
		PKI
	Mitig. Efficiency	Partial removal
		Total removal
		Detection of exploit attempt
	Other	
Mitig. Resolution Method	Manual	
	Automated	
Mitig. RFC #		
Mitig. Start Date		
Mitig. End Date		
Mitig. Log	(string)	
Vulnerability Logs	Date	
	User	
	Note	

The main advantage of IAT is that it captures vulnerabilities that have been analysed by NVAT. This interface could be used to log physical vulnerabilities and exploits, as well.

3.1.4.2.3 Vulnerability Scanner

The vulnerability instances can be detected with COTS vulnerability assessment scanners. JNDMS will collect the instances and link them to their vulnerability definitions (a vulnerability instance should never exist without a vulnerability definition).

3.1.4.2.3.1 Nessus Vulnerability Scanner

Nessus is the industry leading network vulnerability scanner in terms of both adoption and vulnerability definitions (plug-ins). Part of the strength of the scanner is based on an open API for plug-ins allowing the information security community at large to contribute. Plug-ins are written in the Nessus Attack Scripting Language (NASL), an easily readable scripting language. Nessus plug-ins are generally the timeliest, often being released simultaneously with new vulnerability announcements and rarely longer than 24 hours afterward. Also, Nessus plug-ins are frequently released for fringe or non-commercial applications which are not the main focus of commercial vulnerability scanners – this tends to give it greater coverage of potential vulnerabilities on real world hosts.

Many commercial vulnerability scanners include Nessus as part of their scan engine to leverage some of its advantages, and include a number of their own proprietary scanning techniques in addition to Nessus as well as various additional reporting options. Often the weakness in this system is the delay in distributing new Nessus plug-in sets through the commercial channels in a timely fashion.

Nessus scan results can be exported in various formats: .NBE format (machine readable), HTML (human readable), .XML, .NSR format (machine readable - deprecated in favour of NBE), ASCII format (human readable) and LaTeX format (machine readable for publishing, converts to .PDF format). Additionally different Nessus client “front ends” offer different export options.

Table 8 shows a sample scan from a Nessus file:

Table 8: Nessus Scan

Sample	Description
10.163.156.1	IP address or name of the host scanned.
“Vulnerability found on port” or “Warning found on port” or “Information found on port”	What (string)
telnet	Affected service.
(23/tcp)	Port number / Type
The Telnet server does not return an expected number of replies when it receives a long sequence of 'Are You There' commands. This probably means it overflows one of its internal buffers and crashes. It is likely an attacker could abuse this bug to gain control over the remote host's superuser.	Description
For more information, see: http://www.team-teso.net/advisories/teso-advisory-011.tar.gz	Reference (optional)
Solution: Comment out the 'telnet' line in /etc/inetd.conf.	Solution (optional)
Risk factor : High	Risk factors (optional)
CVE : CVE-2001-0554	CVE ID (optional)
BID : 3064	Nessus ID

3.1.4.2.3.2 CA eTrust Vulnerability Manager (eT VM)

eTrust® Vulnerability Manager is a seamless solution that helps the organization know its assets, understand the security exposures and risks, know what action to take, and measure progress in an easily deployable appliance. Using an asset-based vulnerability assessment, eTrust Vulnerability Manager helps to quickly protect an enterprise before its systems are compromised.

eTrust Vulnerability Manager mitigates risk by conducting a scheduled asset inventory where system data is collected through an agent and correlated with validated security knowledge. Through a web-based interface, the organization's risk level is listed in a prioritized task list, allowing targeting the greatest security risk. Remediation instructions are provided allowing taking corrective action. Verification that the resolution worked is also provided to measure progress toward enterprise risk mitigation.

Unlike other vulnerability management solutions, eTrust Vulnerability Manager takes a distinct asset-based vulnerability assessment approach that helps quickly understand what assets are in the environment, along with the exposures to those assets.

It explains how to fix the exposures and validates whether or not the fix has been installed. Furthermore, eTrust Vulnerability Manager implements a seamless vulnerability management process with a single solution.

The eTrust Vulnerability Manager works through deployed agents that have intimate knowledge of the system in which they are installed. These agents feed information back to the eTrust appliance.

Given eTrust VM agents are deployed, system vulnerability data can be exported from the product using the product reporting facilities. It supports reports generated in either Hypertext Markup Language (HTML) or CSV. It should be noted that eTrust Security Command Centre (SCC) is also required to automate the interface and vulnerability events. Otherwise, no automation is possible – only interactive management is possible with eT VM alone.

3.1.4.2.3.3 nCircle IP360

The nCircle IP360 is noted as a vulnerability and risk management system. This system has been identified during the development of JNDMS to be one that DND is likely to deploy.

IP360 is designed for agentless discovery and profiling of large networks. There are a number of ways for information or events to be integrated into JNDMS, including an optional add on to IP360 called the Security Intelligence Hub (SIH).

This tool was integrated into JNDMS using the JSS Client with database access to the SIH.

3.1.4.2.3.4 Centennial Discovery

Centennial Discovery is another tool that has been identified as deployed, in this case at DRDC. Centennial Discovery provides an audit capability to collect and report on software inventory. Centennial can then optionally use the software inventory collected to determine likely vulnerabilities.

This tool was integrated as part of the DREnet deployment efforts using the JSS Client and the Centennial database.

3.1.4.2.3.5 Additional Sources and Transition

The tools chosen to demonstrate key functionality will allow the interfaces to these types of tools to be developed and evaluated. The goal of the TD is to allow other products to be integrated into JNDMS by conforming to the interfaces.

One such initiative that will be monitored is the DND project for Malicious Code Prevention Software. This project is using the Cisco Security Agent.

3.1.5 Safeguards Data

The safeguards are the security measures and controls that prevent or reduce the risk of exploitation of specific vulnerabilities. Table 9 characterizes the safeguards by types, functions and protection types and provides some examples demonstrating their implementation in the CF. Safeguards are described as follows.

3.1.5.1 Safeguard Types

The safeguards are separated into two main categories: the network safeguards and the host safeguards.

3.1.5.1.1 Network Safeguards

These safeguards are part of Network Security. "Network Security is the protection of networks and their services from unauthorized modifications, destruction, or disclosure. Network security safeguards include the application of policies and procedures that include using access control programs, accredited communication devices such as secure routers or hubs, and accredited encryption devices and electronic key encryption protocols." It also includes the monitoring of network operations for security irregularities, as well as assessing the network for vulnerabilities. [R-11]

3.1.5.1.2 Host Safeguards

These safeguards are part of Computer Security. "Computer Security is the protection resulting from measures designed to prevent the deliberate or inadvertent disclosure, acquisition, manipulation, modification, or loss of information contained in a computer system, as well as measures designed to prevent denial of authorized use of the system." [R-11]

3.1.5.1.3 Safeguard Function Categories

The safeguards can also be described by the type of functions they perform. [R-10] describes eight function categories:

1. Correction: implementing a safeguard to mitigate a known vulnerability
2. Detection: detecting the source of a threat
3. Deterrence: discouraging threat agent activity
4. Prevention or Avoidance: changing operations to prevent threat agent activity
5. Containment: limiting the injury/loss that a successful threat event could cause
6. Recovery: providing the ability to quickly recover an IT system from a successful threat event, to its original (or at least to a degraded but usable) state
7. Monitoring: monitoring an IT system for vulnerabilities and/or for threat agent activities
8. Awareness: informing personnel about security issues through well-regulated security awareness programs

For the JNDMS purpose, the following five categories are used:

1. Correction: includes the software patches or configuration changes that must be implemented in order to fix a vulnerability
2. Prevention: safeguards that help prevent threat agent activity (boundary protection, encryption and access control)
3. Detection / Monitoring: a sensor or scanner used to detect vulnerabilities or exploits
4. Avoidance / Containment: includes limiting the impact of a potential incident, or removing any exposure of a known vulnerability
5. Recovery: safeguard providing the ability to recover the original (or degraded) state

Deterrence and Awareness safeguards could be added in the future if pertinent in the defensive posture assessment.

3.1.5.1.4 Protection Types

A safeguard protects (in most cases) the confidentiality, the integrity and/or the availability of the IT infrastructure. Authentication is another type of protection that indirectly protects Confidentiality and Integrity (potentially Availability).

Table 9: Common Safeguards

Types	Safeguards	Safeguard Function Categories					Protection Types				Description of Protection (as per [R-9])	DND Application
		Correction	Prevention	Detection / Monitoring	Recovery	Avoidance / Containment	C	I	A	Access Control		
Network Safeguards (Network Security)	Firewalls or Guards		X	X			X	X	X	X	<p>A <u>firewall</u> is a tool for enforcing the network security policy at the enclave boundary, and has several distinct advantages as a protected network access device. A firewall is generally a protection device between networks communicating at the same security level. A <u>guard</u> is generally a highly assured device that negotiates the transfer of data between enclaves operating at different security levels.</p> <p>The firewall category includes a packet filtering firewall, operating at the network layer, circuit level gateways, operating at the transport layer, application level gateways and multi-level firewalls.</p> <p>By managing the traffic and public access to private networked resources, firewall security partially addresses the issues of integrity, confidentiality and authentication of data that is behind the firewall. Firewalls can also protect against DoS attacks and some can perform traffic encryption and decryption.</p>	<p>DND uses a Boundary Protection System (BPS) to isolate the national system from the CWAN. The BPS is a guard device.</p> <p>Some other firewalls: the DWAN firewalls and internet firewalls, TITAN – BICES firewall, Spartan / Mandrake – Stoneghost firewall, etc.</p>

Types	Safeguards	Safeguard Function Categories					Protection Types				Description of Protection (as per [R-9])	DND Application
		Correction	Prevention	Detection / Monitoring	Recovery	Avoidance / Containment	C	I	A	Access Control		
	Gateways or Routers		X	X			X	X	-	X	<p>A <u>gateway</u> is a centralized computer that provides for communications between information systems. Depending on the product software properties, a gateway may be capable of protocol conversion, audit, and rudimentary access control through Internet Protocol (IP) address recognition.</p> <p>A <u>router</u> is a network layer interface devices that can forward information based on IP addresses, port numbers, and protocol recognition. Routers can have audit capability and can report to firewall or intrusion detection alarm systems.</p> <p>A router configured to act as a firewall is a packet-filtering device that operates at multiple layers and permits or denies traffic through the enclave boundary into the internal network, based on a set of filters established by the administrator.</p> <p>Routers employ internal Access Control Lists (ACL), which can be configured to prevent certain traffic from entering or exiting a network.</p>	Access to the Internet from the Designated Domain through an EAL3 Gateway forms the Unclassified Domain of the DWAN. The DWAN has main router concentrations in Tunney's Pasture and Borden to provide access to National Services. Within Canada there are 32 main routers and approximately 160 Reserve Force and 30 CFHA locations. DWAN extends to 15 fixed locations out of the country, five semi-permanent sites in Bosnia-Herzegovina and up to nine tactical operational deployed locations using the Military Mobile/Satellite Ground Terminals as the interconnecting media to Tunney's Pasture. (The DWAN has approximately 350 routers of various types).

DN0678
Issue 4/2: 06 February 2012

Types	Safeguards	Safeguard Function Categories					Protection Types				Description of Protection (as per [R-9])	DND Application
		Correction	Prevention	Detection / Monitoring	Recovery	Avoidance / Containment	C	I	A	Access Control		
	VPN, Tunnelling, Encryption Devices		X				X	X	-	X	<p>A Virtual Private Network (VPN) creates an effectively private network across a public backbone. The notion of virtual private networking implies an enclave of users who are protected from the network as a whole by some boundary device. Encryption creates secure data pipes (<u>tunnels</u>) between the enclaves. The encryption device can provide data confidentiality, data integrity, peer identification and authentication, audit, and mandatory/discretionary access control services.</p> <p>There are many ways to implement a secure VPN. Layer 3 protection is more affordable than layer 2 and is possible with the use of NES/TACLANE.</p> <p>A tunnel uses a secure protocol, such as IPSec, SSL, etc.</p> <p>Secure Sockets Layer (<u>SSL</u>) exists just above the transport layer and provides security independent of application protocol. This effort has migrated to the Internet Engineering Task Force (IETF) as the Transport Layer Security (TLS) protocol, which provides data encryption, server authentication, message integrity, and optional client authentication for a TCP/IP connection between two applications.</p>	<p>DND has approximately 80 CNES (Canadian Network Encryption Systems) and approximately 400 TACLANEs (TACLANE replaces CNES).</p> <p>The <u>CNES</u> security platform is a Type 1 Network Encryption Security Device designed to Secure Data Network System (SDNS) standards and endorsed by CSE to handle all levels of GoC classified and designated information, up to Top Secret level. The CNES is software configured and keyed using a STU-III-like method. KSD-64A/PKA64KC keys are supplied through CSE Electronic Key Management System (EKMS).</p> <p><u>TACLANE</u> is an IPSec and ATM encryptor that supports a variety of IP and ATM network configurations. TACLANE has a KSD-1 Crypto Ignition Key. TACLANEs can be used to partition existing networks into Secure Virtual Networks.</p> <p>The DWAN Classified Domain consists of Secure LANs protected by CNES/TACLANE, connected through the Designated Domain.</p>

DN0678
Issue 4/2: 06 February 2012

Types	Safeguards	Safeguard Function Categories					Protection Types				Description of Protection (as per [R-9])	DND Application
		Correction	Prevention	Detection / Monitoring	Recovery	Avoidance / Containment	C	I	A	Access Control		
											<p>Internet Protocol Security (<u>IPSec</u>) is the security framework standardized by the IETF as the primary network layer protection mechanism. IPSec consists of two parts: an authentication header (AH), whose purpose is to bind the data content of IP frames to the identity of the originator, and an encapsulating security payload (ESP), for privacy. The AH is intended for use when integrity of information is required but privacy is not. ESP is intended for use where data confidentiality is required.</p>	<p>CNES/TACLANE provide a secure connectivity between a MCOIN or AFCCIS client and the Classified Network (CNET).</p> <p>Another example is the Deployable DWAN Equipment (DDE) that connects to the classified Deployed Routers in Tunney's Pasture through a link secured by CNES.</p> <p>DND also has the Defence Virtual Private Network Infrastructure (DVPNI). This project allows users to securely connect Government-issued remote workstations to the DWAN and allow the secure exchange of Protected B Designated Information on the DWAN. The DVPNI project has two phases: the first phase is Secure Remote Access (SRA) and the second phase is Secure Network Infrastructure (SNI) (not yet started).</p>

DN0678
Issue 4/2: 06 February 2012

Types	Safeguards	Safeguard Function Categories					Protection Types				Description of Protection (as per [R-9])	DND Application
		Correction	Prevention	Detection / Monitoring	Recovery	Avoidance / Containment	C	I	A	Access Control		
	Secure Remote Access (also a VPN)		X				X	X	-	X	Remote access implies a sole user gaining access to the enclave by some protected means. Although the mechanisms to implement this access may be similar to that used for VPN, the details of the connection are vastly different. Remote access through an Internet Service Provider (ISP) consists of simultaneous connections to a private entity and a public entity without any intervening firewall.	DND has VPN Remote Access to DMZs.
	Authentication Servers		X	X			-	-	-	X	An Authentication Server is a system that provides authentication services to other systems on a network. The server issues a cryptographic ticket, which certifies the identity of its owner. Tickets are usually time-expired. The architecture of a Trusted Third Party (TTP) authentication uses a third entity, trusted by all entities, to provide authentication information.	

DN0678
Issue 4/2: 06 February 2012

Types	Safeguards	Safeguard Function Categories					Protection Types				Description of Protection (as per [R-9])	DND Application
		Correction	Prevention	Detection / Monitoring	Recovery	Avoidance / Containment	C	I	A	Access Control		
	IDS			X			-	-	-	-	The network intrusion detection systems identify attacks, misuse, or abuse of computer systems. Effective intrusion detection systems detect both insider and outsider attacks in near real time. The objective of these systems is to detect malicious and unintended data and actions (e.g., altered data, malicious executables, requests that permit unintended resource access, and unintended use of intended services) by scanning network traffic for known "attack signatures" using a rules-based engine. "An attack signature can be any pattern or sequence of patterns that constitutes a known security violation." The type of protection is not applicable.	
	Vulnerability Assessment Scanners			X			-	-	-	-	The type of protection is not applicable.	
Host Safeguards (Computer Security)	Software patches or upgrades, or configuration change	X					-	-	-	-	These are the solutions typically associated with software vulnerabilities. The type of protection is not applicable; it removes a vulnerability completely.	

DN0678
Issue 4/2: 06 February 2012

Types	Safeguards	Safeguard Function Categories					Protection Types				Description of Protection (as per [R-9])	DND Application
		Correction	Prevention	Detection / Monitoring	Recovery	Avoidance / Containment	C	I	A	Access Control		
	Identification and Authentication (I&A) (password strength)		X				-	-	-	X	I&A provides effective access control. Access control means limiting access to networked resources (hardware and software) and data (stored and communicated). The goal of access control is to prevent the unauthorized use of these resources and the unauthorized disclosure or modification of data.	
	Service / software with access control		X				-	-	-	X	This is a software or user service requiring user authentication.	
	File encryptors		X				X	X	-	X	<p>Encryption allows a user to encode information so that only specified users can decode the information (considered an implementation of discretionary access control) without an unreasonable amount of effort.</p> <p>The <u>file encryptors</u> provide confidentiality and integrity for individual files, provide a means of authenticating a file's source, and allow the exchange of encrypted files between computers. This protects individual files but does not protect all of the files on the drive. File encryptors are applicable to security-enabled applications.</p>	DND can encrypt and decrypt files using the Entrust PKI. The Entrust client software (Entelligence) enables electronic processing of accountable unclassified and designated message traffic up to Protected B.

DN0678
Issue 4/2: 06 February 2012

Types	Safeguards	Safeguard Function Categories					Protection Types				Description of Protection (as per [R-9])	DND Application
		Correction	Prevention	Detection / Monitoring	Recovery	Avoidance / Containment	C	I	A	Access Control		
	Service / software with encryption		X				X	X	-	-	Applications that use a standard Cryptographic Application Programming Interface (CAPI) can access an external cryptographic service.	The Secure Common Email (SCEM) uses the PKI services** for email encryption and digital signatures.
	Media encryptors		X				X	X	-	-	The <u>media encryptors</u> protect the confidentiality and integrity of the contents of data storage media. They can also perform a role in maintaining the integrity of the workstation by verifying the Basic Input/Output System (BIOS) and ensuring that configuration and program files are not modified. Media encryptors need to leave some system files unencrypted so that the computer can boot from the hard drive. Most of these files can have their integrity protected by a cryptographic checksum; this will not prevent a tamper attack but will alert the user that the data has been altered.	
	Signature checksum			X			-	X	-	X	Signature checksum provides data integrity by digitally signing data. Typically, the data requiring protection is used to calculate a smaller value, such as a parity, checksum, or hash. This value can then be digitally signed.	A DND user can sign files with its digital signature, using the Entrust PKI client (Entelligence).

DN0678
Issue 4/2: 06 February 2012

Types	Safeguards	Safeguard Function Categories					Protection Types				Description of Protection (as per [R-9])	DND Application
		Correction	Prevention	Detection / Monitoring	Recovery	Avoidance / Containment	C	I	A	Access Control		
	Security features of the OS environment		X				X	X	X	-	The Windows security settings include authentication, access control, permissions, automatic updates and security levels for different zones.	
	Redundancy *				X		X	X	X	-	Redundancy is to ensure that an alternate system or method is implemented to provide a given service or information. In the event that a system has been compromised, authorized traffic can be diverted (seamlessly) to the alternate system. The alternate system should have a different configuration to ensure that it is not vulnerable to the same exploits affecting the primary system (and vice-versa).	
	Backup Protection				X		-	X	X	-	A backup is a copy of system files produced on separate media so that the system can be regenerated in the event of a crisis.	In the Canadian Forces, every classified information system must be backed up periodically.
	Network connectivity (on or off)					X	-	-	-	X	Temporary solution to avoid exposure to external threat. Disconnecting a workstation from the network would completely protect the "remotely exploitable vulnerabilities" on the workstation from external attackers.	

DN0678
Issue 4/2: 06 February 2012

Types	Safeguards	Safeguard Function Categories					Protection Types				Description of Protection (as per [R-9])	DND Application
		Correction	Prevention	Detection / Monitoring	Recovery	Avoidance / Containment	C	I	A	Access Control		
	Virus Scanning	X	X	X			-	-	-	-	<p>Virus detectors can be used to protect a network or an individual client from malware. Proper virus scanner configuration must be maintained. Protection typically is provided at two places in the architecture: at the gateway and at workstations that access information services.</p> <p>Virus scanning protects against vulnerabilities related to malware. The CIA types of protection are not applicable.</p>	
	Spyware / Adware Scanning	X	X	X			X	-	-	-	<p>Spyware or adware scanners can protect a network much the same way that virus scanners work.</p>	
	HIDS			X			-	-	-	-	<p>The host intrusion detection systems protect individual client platforms.</p> <p>The type of protection is not applicable.</p>	
	HIPS		X	X			-	-	-	-	<p>The host intrusion prevention systems monitor and intercept system calls to the kernel or APIs in order to log and prevent attacks.</p> <p>The type of protection is not applicable.</p>	

Types	Safeguards	Safeguard Function Categories					Protection Types				Description of Protection (as per [R-9])	DND Application
		Correction	Prevention	Detection / Monitoring	Recovery	Avoidance / Containment	C	I	A	Access Control		
Physical Security	Access Control		X	X			X	X	X	X	Physical security includes tangible security mechanisms, such as guards, locks, and fences. The idea is to build a physically secure enclave.	<p>DND assigns physical security zones to equipment according to the sensitivity of the information on the asset.</p> <p>High-Security Zone – TOP SECRET with Special Compartmented Information (SCI).</p> <p>Security Zone – Protected C and TOP SECRET.</p> <p>An area to which access is limited to authorized personnel and properly escorted visitors. Accessible from an Operations Zone, and through an entry point.</p> <p>Operations Zone – Protected A, B, Confidential and Secret.</p> <p>An area to which access is limited to personnel and to properly escorted visitors.</p> <p>Reception Zone or Public Access Zone – No sensitive processing permitted.</p>

*Redundancy

The type of redundancy considered in this TDP will be similar to what's done in IAT: "the redundancy is strictly defined as the existence of another piece of equipment providing the same user services, at the same locations, and that is not impacted by the incident".

**Public Key Infrastructure (PKI) Services

PKI is a unique category because it does not directly satisfy the security requirements: it facilitates the use of security building blocks needed to support the other technical safeguards. The PKI architecture is essential in any network security solution. In JNDMS, it will be captured as a service. The safeguards using this service will be linked to the required PKI services.

A PKI is a set of systems for managing paired public and private keys, which principals (including users and systems) can use to authenticate to each other, to sign documents and to send private messages to each other. PKI enables encryption, digital signature and authentication capabilities to be consistently and transparently applied across a broad range of applications and platforms. A PKI can reduce the costs of managing and distributing encryption keys among large numbers of users. The PKI is an enabler: it forms building blocks used by other security technologies (VPN, network and remote access).

3.1.6 Security Events

The security events will be sensed through a set of sensors in the CND environment (mainly IDS and Firewalls). The security events may also be captured in the organization's trouble ticketing tool. The vulnerability instances are another form of security events. They are discussed in the Vulnerability and Exploit Data section (3.1.4).

3.1.6.1 COTS Selection

3.1.6.1.1 Security Sensors

During the JNDMS TD, event data will come from simulated CND data sources. This will be achieved by harvesting representative data samples from a production network and then "replaying" them in the lab as required during system testing. Permission should be sought by the DRDC Ottawa Network Information Operations (NIO) section to use data generated on the Defence Research Network (DREnet) as it would provide us with a variety of interesting data sources and traffic samples.

It is important to note that a given set of data samples taken from a production environment will not necessarily include interesting security data, but it will, regardless, serve other purposes:

- It will represent environmental "noise" events
- It will indicate of typical event volume generated by a given device type in an environment with known characteristics
- It will show the log format and data fields provided by each kind of device in question
- It will give a suggestion as to the variety of event types generated by the device

During the course of the in lab trial, it will be necessary to generate specific kinds of attacks or anomalies on top of the simulated "noise" traffic. This data can be handcrafted on demand using "live data sources" as detailed in section 3.3.3 (System Interfaces for the Security Information Management). This approach will allow us to carefully place the "needles" in the "haystack" as the correlation rules are being validated.

The system interface for the "haystack" data in the lab deployment would be "pre-recorded" data files, stored on the ISM server, and copied into the appropriate "inbox" (location in ISM file hierarchy) as needed. New files in an inbox are monitored and processed by the ISM rules engine as default behaviour.

Data samples would be recorded from some of the sources outlined in Table 10, using the mechanisms listed beside each data source.

Table 10: Possible Sources of Security Events

Sources	Interface
Cisco Secure IDS	syslog
Snort IDS	syslog
Nessus	SCP
Cisco IOS	syslog
Checkpoint Firewall NG	OPSEC LEA or syslog
Nortel Contivity VPN gateway	syslog
Unix Services <ul style="list-style-type: none">• Apache reverse proxy• Simple Mail Transfer Protocol (SMTP) server• Mail gateway anti-virus log [AMaViS]• IPtables [firewall logs]	syslog
IIS logs (MS webserver)	W3C extended logging
FreeRadius (dialup authentication)	mySQL database
SNIPS (open source service monitoring)	flat files – requires research

3.1.6.1.2 Ticketing Tool

The organization opens a ticket every time an incident affects their network. DND currently uses Remedy as their ticketing tool but it does not capture the information in a consistent fashion or with enough precision for the required correlations. The tickets will be simulated in this TDP and Remedy will serve as a baseline for comparison. The data fields available in the Remedy ticketing tool are listed in Table 11.

Table 11: Trouble Ticket Attributes

Field	Field Value
Ticket ID	TTXXXXXXXX
Opened Date	(date)
Closed Date	(date)
Status	Assigned
	Closed
	Hold
	Open
	Resolved
	WIP
Short Description	(string)
Network	(list of network)
Asset	(list of assets)
Operational Importance	1- None
	2- Minimal
	3- Moderate
	4- Significant
	5- Critical
Trouble Type	Bell Scheduled Outage
	DND Scheduled Outage
	Investigation
	Non-TSRP
	R&O
	Security Incident
	Service Degradation
	Service Outage
	TSRP - Other
Service Delivery Access Point	(list of locations)

3.2 Enterprise Infrastructure Management

This section discusses the overview of the EIM, the selection of the tools used to manage the enterprise information and how this subsystem interacts with other subsystems.

3.2.1 Overview

The EIM component will be created to manage IT infrastructure devices and events and will use the information from those devices and events to feed the JNDMS Data Warehouse (DW) and Decision Support System (DSS) with relevant information about the discovered infrastructure elements and their state.

3.2.2 COTS Selection

The following components are included as components of the Enterprise Information Management.

- Unicenter Network and Systems Management (NSM)
- Unicenter Asset Management
- Spectrum
- eHealth
- Centennial Discovery

Table 12 summarizes the the role of each component within JNDMS:

Table 12: EIM Component Roles

Component	Role	Notes	Final Build
Unicenter NSM	Console and event management	The event console within Unicenter is used to consolidate the EIM component event flows and is used for the communication between EIM and JNDMS core.	N
Unicenter Continuous Discovery	Live Discovery	Unicenter has the ability to monitor the network and periodically report on discovered assets. This can work in corporation with the	N

DN0678
Issue 4/2: 06 February 2012

Component	Role	Notes	Final Build
		Spectrum discovery to provide better coverage.	
Unicenter Asset Management	Software asset discovery and reporting	This component can query and collect what is installed on servers and workstations and report.	N
Spectrum	Discovery	Spectrum is used as the primary component for discovery. The discovery module within Spectrum gives detailed, port level, information on network connections.	Y
Spectrum	Availability Monitoring	Spectrum will continually monitor the availability of the assets that it has discovered. Spectrum will report on the status of these assets, as well as perform some root cause analysis to determine the asset most likely causing groups of issues.	Y
Spectrum	Software asset reporting	Spectrum uses SNMP and can use information stored in MIBS to report on installed software.	N
Centennial Discovery	Software Asset inventory	This tool can use local or domain credentials to perform the inventory of software.	Y
eHealth	Performance Monitoring	eHealth is used to report on performance issues. This component reports through NSM.	N

3.2.3 System Interfaces

The system interfaces to the IT infrastructure will support the Discovery and Availability Monitoring IT infrastructure functions. Both functions will primarily use the same interface mechanism to interact with the IT infrastructure: Internet Control Message Protocol (ICMP) ping and SNMP get requests.

From a discovery perspective, which will be done on a regular basis to ensure an accurate perspective of the IT infrastructure, all infrastructure elements will be identified and new (or changed) devices will be highlighted through specific IT change events.

The Spectrum and Unicenter NSM Continuous Discovery components will be used to achieve this function.

In addition to the ICMP (ping) and SNMP interfaces, additional discovery mechanisms may be used. These may include (assuming additional conditions are met):

- Routers' Address Resolution Protocol (ARP) cache (assuming the routers' SNMP read-only community string is known),
- Dynamic Host Configuration Protocol (DHCP) traffic monitoring (assuming DHCP traffic is sent to the NSM manager by the network routers), and
- Traffic analyzer (assuming the discovery agent is connected to the segment's mirrored port).

It is important to note that it is the relationship between the name and the Internet Protocol (IP) address of a device that is the key to its unique identification.

In addition to pure discovery data, additional location and relationship data must also be manually added to the CMDB. This critical additional data will have to be inserted somehow into the data flow to complement the information provided by the discovery mechanism.

The JNDMS will treat the CMDB as a data source, but not manage this data. The collection and management of this dataset is outside of the scope of JNDMS. The JNDMS must ensure that changes to the CMDB are provided.

From an availability monitoring perspective, all discovered IT infrastructure elements will be actively monitored by Spectrum and Unicenter NSM and any failed network device or infrastructure server will also be highlighted through specific IT availability events.

3.2.4 Subsystem Interfaces

From an IT infrastructure data flow perspective, several types of data will be shared from EIM to JNDMS:

1. IT infrastructure change events, informing JNDMS of all IT infrastructure changes
2. IT infrastructure availability events, highlighting IT infrastructure failures
3. IT Asset (software) change events
4. IT infrastructure performance events

Each type of data will be handled in the same way. They will be automatically forwarded to the JNDMS Services through specific event actions (given it is critical that all IT infrastructure events be brought up to JNDMS attention immediately). In that case, it will be up to the NSM event management interface to push the relevant event data into the JNDMS Data Warehouse.

In either case, the JNDMS DSS must be triggered following the push of the event data to JNDMS by the NSM Event Management component. The NSM event management will trigger a JNDMS agent that will send the event into the JNDMS Services. It is then up to the JNDMS core components to handle the new event.

The initial JNDMS agent that will integrate with the NSM event management is the JSS Client (see section 3.4.3, JSS Client).

Spectrum and eHealth make use of integration into the Unicenter family of products. Both of these products will report events through the NSM Event Management console. The integration of Spectrum into JNDMS, through NSM, involves a number of integration scripts and tools. Figure 3-2, below shows the communication flows between these tools.

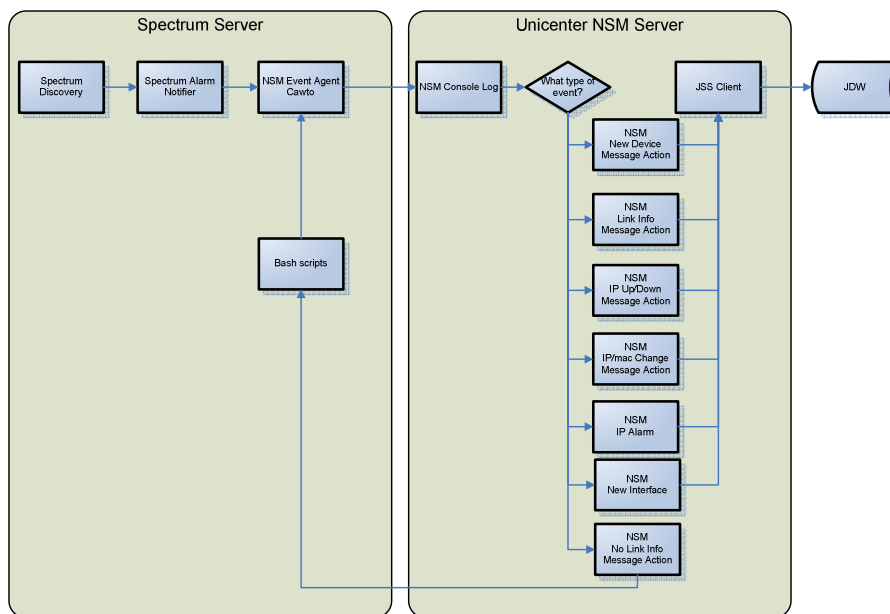


Figure 3-2: Spectrum and Unicenter Communications

Centennial Discovery makes use of periodic queries to the Centennial database. This solution is very flexible and enables JNDMS to get the current state of the software inventory and manage updates or changes to the inventory. This option is, however, dependent on the schema used by Centennial. Each version of Centennial may have to be separately supported.

3.2.4.1 IT Infrastructure Baseline

As defined in section 3.1.1, another concept is also required to properly manage the IT infrastructure events: the availability of an IT infrastructure baseline. This baseline will allow JNDMS to make sound decisions on the impact of an IT infrastructure change or availability event.

The baseline should include not only the list of devices that are known to exist in the IT infrastructure but also the devices' location and the relationships between the devices themselves and the IT infrastructure and services they support.

The IT infrastructure baseline should be created and must be maintained manually, from known valid infrastructure information (similar to the one that exists in the DND National Telecommunication Management System [NTMS]). However, initially, the asset inventory gathered from the EIM discovery tools could be used to create the initial JNDMS baseline inventory.

In addition to the discovered information there is potentially additional information that could, or should, be managed through the EIM components. Two examples would be location information and point of contact information.

The location information is currently managed through the operation reporting interfaces of the JSS, and not through the EIM. It has been noted that either component could potentially provide location information.

The current design will allow for EIM components to report on POC as well as additional data sources such as a POC database.

If required, to support that initial inventory load, additional data fields can also be created into the appropriate devices to properly insert the manual information about location (i.e., coordinates, physical address, floor and/or floor location) and points of contact.

3.2.5 Data Storage

The Unicenter NSM EIM product manager, Spectrum and eHealth have their own database to host its own dataset. The NSM EIM product manager as the core of the JNDMS EIM has a unified database called the MDB.

The CA MDB hosts all information about all new versions of all CA products. It is CA's direction to ensure that all products will use the CA MDB in the future.

The current version of the MDB is used by all the new r11 Unicenter products (including NSM).

Its schema is published and available as part of the MDB documentation. This schema has been used to validate the core JNDMS data model.

From the information available in the published schema definition, the necessary data exchange between NSM and JNDMS will be readily achievable.

It should be noted, however, that events are not saved in the MDB. They are saved in proprietary binary files on the NSM manager server. Using CA standard event management commands, any and all parts of the events flat file database can be exported into JNDMS through an agent, as required. From that file, the JNDMS Data Transformation component could be used to import events in bulk into the JNDMS Data Warehouse, again if required.

The Centennial Discovery product has its own database to store all of the software inventory. This database is queried directly by JNDMS.

3.2.6 Rules

The EIM product manager (Unicenter NSM) has its own ability to identify relevant events. Through standard NSM message records and actions, significant change and availability events will be chosen and redirected to the JNDMS Data Warehouse.

The rules will, therefore, involve identifying the significant events, code these events into the NSM message record database, and then code the relevant action to push the event to JNDMS.

Relevant events will be:

1. Change events, sent to event management through the regular network discoveries
2. Availability events, sent to event management through the continuous network monitoring
3. Performance alerts

During the DREnet deployment the rules within NSM were not used. Spectrum and Centennial were the primary EIM data sources.

3.2.7 Integration and Transition

The role of this Technology Demonstration is not to emulate the final configuration of what the DND network will make use of, but to provide a framework to integrate various COTS products where they exist. This is the reason that the CA suite of tools had been originally chosen. During the effort of the DREnet deployment products were chosen based on products existing within DRDC and products that DRDC had plans to deploy and manage in the long term.

3.3 Security Information Management

This section describes the SIM subsystem. This includes an overview of the component, the tools that make up this subsystem, and the relationship between this subsystem and other JNDMS subsystems.

3.3.1 Overview

The SIM component will process data inputs from the CND environment. From this data stream, significant security events and other events of special interest will be injected into JNDMS. In order to achieve this, the Information Security data must be efficiently acquired, normalized, correlated and stored in a secure data warehouse for future forensic analysis.

3.3.1.1 Component Description

SIM solutions enable security teams to rapidly and comprehensively identify information security incidents, to deploy resources on the threats that pose the greatest risk to the business, to assess and resolve these incidents with the strongest security team productivity and capability, and to affordably scale security coverage enterprise-wide.

Capabilities or features of the SIM component of the JNDMS TD include:

- Acquire security event data such as logs, alerts, system events and formatted reports, from various sources
- Pre-process and store security event data
- Filter, aggregate, de-conflict, and consolidate security events
- Normalize security event data to a common, complete and consistent format
- Acquire and correlate Vulnerability Data with security events
- Prioritize individual security events on a normalized severity scale, including supporting evidence of severity assessment
- Summarize the severity of the overall security situation, of on-going events, and of multiple-simultaneous events including supporting evidence of severity assessment
- Assess the effect of security events on the CND environment

3.3.2 COTS Selection

The Intellitactics Security Manager (ISM) was selected as the SIM product for the JNDMS TD. ISM replaces NSM as the core SIM product offering from Intellitactics.

Criteria for selection of a COTS product for the SIM component include the ability to achieve the abovementioned capabilities, plus the following:

- Preference for a Common-Criteria certified solution
- Aversion to the need for active host-based agents
- A mechanism to validate data integrity as part of a JNDMS data repository
- Interoperability with the existing DND SIM implementations (Intellitactics' NSM)

Notably, the DND Computer Incident Response Team (CIRT) team has a large investment in their deployed ISM infrastructure, which has been in production for several years, and has a great deal of customization specific to the needs of the Canadian Forces Network Operations Centre (CFNOC). Additionally, the solution needed to meet all of the following standard SIM criteria:

- Supports data acquisition from multiple vendors/ multiple classes of security point products
- Provides a common management interface for all types of logs/ alarms
- Normalizes security point product data to uniform format
- Correlates security point product data with a mature correlation engine, containing out-of-the-box rules, including the ability to:
 - Aggregate or rollup similar events
 - Filter known unimportant events
 - Prioritize events
 - Prioritize alarms based on asset significance
 - Prioritize alarms against vulnerabilities of assets
- Provides reports suitable for post-incident analysis, management, and trending
- Processes data and send alarms in near real-time
- Capable of sending customized alerts and triggering custom actions

Additionally, the Intellitactics solution also had the following unique set of features that are useful to the JNDMS research project:

- Avoids the use of Agents
- Has achieved Evaluation Assurance Level (EAL) 2 certification under the Common Criteria scheme

- Has graphical user interface (GUI)-based configuration and customization
 - Correlation/ Rule engine has a visual editor to develop rules
 - Has a GUI which allows rapid development of custom device support and customization of existing device support
 - Has a GUI to configure database for data retention and data aging policies
- Has a hierarchical classification (taxonomy) for security events
 - Event taxonomy allows for filtering of event types at several levels of granularity (for example: ids.* or ids.detect.exploit.* or ids.detect.exploit.web.*)
 - Has a standardized numeric priority for each taxonomy type (similar to the “Base Metric” in CVSS)
- Has customizable dynamic visualizations/ representations of security events and network behaviours allowing:
 - Visualization of relationships between attackers and targets
 - Visualization of threats and anomalies
 - Replay and visualization of historical events of a given attacker or attack type
 - Layering of data from multiple views or multiple sources into one

These relationships are held in state inside the rule engine, in the form of directed graphs, and can be queried by the system.
- Has an embedded data warehouse
 - Data warehouse supports three tiers of data, including the retention of compressed raw security data in its original unprocessed format, the post-processing normalized logs compressed, and aggregated summary data in a optimized relational database
 - Key meta-data about the warehouse itself is stored including integrity validation information, such as the message digests of all raw log files, timestamp and disk location information, and both top down and bottom up linkages between correlated events in the summary back through the normalized logs, to the original unprocessed raw logs
 - Has fast query response times, high-speed data insertion, retention of original logs for forensic purposes, and small storage footprint compared to both raw log files and relational databases
 - Ensures ease of maintenance through GUI-based configuration of granular data retention and aging policies
- Has a flexible and sophisticated rules engine
 - Permits viewing and modifying rule engine working memory (state) in the GUI
 - Supports self modifying rules
 - Has documented API to add new rule operators
 - Has the ability to use custom java in an operator
 - Has the ability to use external system calls in an operator

3.3.3 System Interfaces

This section describes the SIM interfaces to current data sources, how to extend those data sources and interfacing to test data.

3.3.3.1 Interface to Test Data

A simple, uniform interface can be used to enter test data into ISM. Representative data samples are held as files on the ISM server and copied into the appropriate “inbox” (directory) as needed. They are then automatically picked up by the ISM rule engine and processed. This approach has the key advantage of ease of implementation; however, it also has the disadvantage of processing all of the “replayed” events at one time, rather than over a long period of time. All timestamps are stored and harvested from the original source data, rather than the time the files were copied into the inbox, which offsets the effects this might have on various correlation rules.

3.3.3.2 Extending Current Data Sources

Currently, not all Intrusion Detection Systems provide a CVE ID with a reported alarm (Snort being a notable exception). Typically, the alarm will have a “signature ID”, which can be used to link back with the product’s knowledgebase where the CVE ID can be found (as is the case for Cisco IDS).

For JNDMS, we will require IDS/ vulnerability correlation based on CVE ID, and therefore, we must extend the data reported by the IDS to include the CVE. To limit the effort involved in doing this mapping for thousands of signatures, a subset of signatures will be used for the purpose of the TD. Research will have to be conducted to find the best place to do this from possibilities including modification of sample data sets or lookup tables stored in the SIM. Some issues in performing this mapping for the purpose of correlation will include multiple CVE numbers per signature – this can occur when a signature is written to detect an exploitation method rather than a vulnerability instance. Additionally, not all signatures will have a CVE (and cannot have one assigned) – if they are not exploit centric for example.

The vulnerability scanners already report CVE IDs with each vulnerability instance detected, and require no changes. As with IDS signature to CVE mappings there are some data integrity issues, as well – some vulnerabilities involve multiple or no associated CVE entries. Also some vulnerabilities involve “compound” CVE entries to be exploitable – so a CVE match between an IDS alarm and a vulnerability instance will not necessarily mean that successful exploitation has occurred.

3.3.3.3 Interfaces to Live Data Sources

The system interfaces of the SIM as they pertain to JNDMS exclusively focus on the capability to collect data from the CND environment. SIM has “push” and “pull” type interfaces.

3.3.3.3.1 Push Interfaces

Push interfaces are those in which data sources in the CND environment explicitly export events or logs to the SIM, which processes them upon receipt.

Push interfaces can be attached using Out Of Band (OOB) or in-band connections. OOB security networks allow for the isolation of security data from the production network. OOB connection is not a vendor-supplied feature (today), but is accommodated for by the Intellitactics architecture. In the context of JNDMS this would be used to push data from a lower classification network to a higher classification network. A suitable guard technology that ensures isolation of the two networks (such as a one-way diode) would have to be selected or simulated for the TD.

Push interfaces include:

Syslog

(RFC 3164 - <http://www.faqs.org/rfcs/rfc3164.html>). The syslog interface to ISM is syslog-ng (http://www.balabit.com/products/syslog_ng/), which allows User Datagram Protocol (UDP) and Transmission Control Protocol (TCP) delivery of UNIX syslogs on arbitrary port numbers. In the default configuration this includes listening for syslog formatted data on 514/UDP and 514/TCP.

SMTP

The Simple Mail Transfer Protocol as defined by the Requests For Comments (RFC) 821 (<http://rfc.net/rfc821.html>) and 2821 (<http://rfc.net/rfc2821.html>). By default ISM will handshake and receive SMTP messages on 25/TCP.

SNMP Traps

The SNMP (<http://www.faqs.org/rfcs/rfc1157.html>) interface to ISM supports receipt of SNMP traps using versions 1, 2, and 2c of the protocol, and defaults to 162/UDP.

Intellitactics LMP

Upstream Intellitactics components (servers or agents) can send data according to the Lightweight Message Protocol (LMP), a clear text protocol that typically is sent on TCP ports in the 9300-9400 range. LMP can also be carried over Secure Sockets Layer (SSL) encrypted sockets in the same TCP port range.

Proprietary Security Data Protocols

Leading security protocols, such as RDEP, RADIUS, and OPSEC LEA, can be used to push event and log data to ISM servers.

Custom

Non-standard sources of data can also push to ISM servers. The key criterion for successful delivery and subsequent processing is being able to deliver the data to the appropriate “inbox” on the ISM server – each security product must have a unique directory in the ISM file hierarchy to accommodate incoming data.

3.3.3.3.2 Pull Interfaces

Pull interfaces are those in which data sources in the CND environment have no usable facility to export events or logs, and they must be explicitly harvested either periodically or through external triggers so they can be processed by the SIM. The Intellitactics architecture explicitly avoids the use of host-based agent software where feasible when pulling data from the CND environment.

Pull interfaces require in-band connections to the CND environment. Isolation needs to be achieved through guard technologies deployed between the in-band and out-of-band Intellitactics servers. For example, a database server containing security information must be queried using a transaction from an in-band system. This in-band system turns around and pushes the acquired data up through a one-way link to the OOB system.

Pull interfaces include:

File Transfer Protocol (FTP)

File Transfer Protocol as defined by RFCs 959 and 1579. FTP can be used to pull log files from a remote server and deliver them locally to an inbox on the ISM server.

Secure CoPy (SCP)

SCP is a subset of the functionality of Secure SHell (SSH), which allows secure file transfer over an encrypted TCP connection. SCP can be used to pull log files from a remote server and deliver them locally to an inbox on the ISM server.

File Shares

Server Message Block (SMB) or Network File System (NFS) shares can be monitored over the network by ISM for new files or updates to existing files, and the deltas can then be processed.

JDBC

JDBC or ODBC can be used to monitor an almost arbitrary variety of relational databases that store data. ISM can then poll the database in question at specified intervals and process the deltas. This is achieved by keeping track of a cursor into the database at the position of the last data acquired by the Intellitactics server and only querying records newer than the cursor position.

Flat Files

Flat or formatted text files can be monitored by ISM and the deltas can then be processed.

3.3.3.4 System Interface Design

ISM incorporates a technology called the Device Modeling Framework (DMF) to process event and log data. Each individual type of log file or event stream is modeled with an instance of a DMF called a Device Modeling Framework Definition (DMFD). New DMFDs can be created from scratch using the GUI or existing DMFDs can be altered to accommodate potential customization of logging formats by security products. This gives ISM a highly flexible system interface in terms of data inputs from the CND environment.

3.3.4 Subsystem Interfaces

The system interfaces of the SIM as they pertain to JNDMS focus on the methods used to inject and receive data through the JNDMS System Services component.

3.3.4.1 Security Incidents

The security incidents are propagated to the JNDMS core by pushing these events through the JNDMS System Services component. The Intellitactics tool can be extended through a Java API. The ISM will convert incidents to XML and deliver them to JNDMS System Services using SOAP over Hypertext Transfer Protocol (HTTP).

3.3.4.1.1 ISM Feature Types

ISM categorizes events into Features. A Feature is a logical definition of the characteristics or patterns reported in the security data. ISM threat analysis detects or deduces "Features" from the input data sources. ISM takes incoming event data, analyzes it, and then uses it to build a description of the behaviours and characteristics - the Features - of each IP address. The composite description, known as a Feature Map, holds the key attributes that describe behaviours or characteristics of an IP address. A Feature may be one indication of a Security Incident (but it may also be benign). It requires further analysis to determine the cause.

Table 13 lists the categories of features.

Table 13: Security Feature Categories

ID (for DSS)	Feature	Description
1	anomaly	Shows involvement in anomalous or atypical network activity.
2	auth	Shows involvement in authentication or authorization activity, such as login or logout activities on your network and for specific assets. Types of auth activities include the following:
2A	auth_deny:	Shows that an authorization or authentication request has been denied.
2B	auth_expire	Shows that an authorized connection or session has expired, probably because there was no transmission of data (the session was idle).
2C	auth_force	Describes an authorization decision that was not in direct response to a request (for example, a forced logout that was the result of a timeout or an account disabling).
2D	auth_grant	Shows that an authorization or authentication request has been granted.
2E	auth_redirect	Shows that the authorization or authentication request has been directed to a third-party authorization or authentication system. Usually, this means that the external gateway passes credentials to the authentication system for authorization to establish a session with the requesting party.
2F	auth_reply.	Shows the reply to the initial authorization or authentication request.
2G	auth_request	Shows the initial authorization or authentication request.
2H	auth_shun	Shows authorization or authentication requests that are shunned because of some criteria (for example, a specific IP address block).
3	compromise	Shows involvement in a compromise. In a compromise, an unauthorized access to a host is obtained through exploitation of a vulnerability on that host. If a host has the Feature Attribute of e_s_compromise, it suggests that the IP address is compromised. The source is not the attacker; rather it is the host that was generating evidence of compromise. If a host has the Feature Attribute of e_t_compromise, it suggests that the IP address attacked another host and compromised it. The source is not the attacker; the source is the host generating the evidence of compromise (the host that was compromised).
4	conf	Shows that a configurable attribute of a service has been changed.
5	disclose	Shows that a service answered a request for information that might be useful to attackers.
6	dos (denial of service)	Shows involvement in an attempt to deny service. An IP address with the Feature Attribute of e_s_dos has been seen subjecting other IP addresses to DoS attacks. A Feature Attribute of e_t_dos suggests that the IP address has been the target of a DoS attack.
7	error	Shows involvement in error conditions that can be of any severity. This is usually meta data about the sensor or safeguard – the device has thrown some kind of internal error.
8	evade	Shows that a request was made to a service that suggests that the client is attempting to evade detection by monitoring systems.
9	exploit	Shows involvement in an attempt (successful or not) to compromise another

ID (for DSS)	Feature	Description
		system. An IP address with the Feature Attribute of e_s_exploit has been seen attacking others. An address with a Feature Attribute of e_t_exploit suggests that the IP address has been attacked.
10	info	A default Feature type that captures information that is not necessarily noteworthy on its own but can be important when viewed in conjunction with other Feature information.
11	insecure	Shows involvement in insecure behaviour. An IP address with the Feature Attribute of e_s_insecure has been seen behaving in an insecure way. A Feature Attribute of e_t_insecure suggests that the IP address has been involved in insecure behaviour but did not necessarily initiate it.
12	misuse	Shows involvement in activity normally treated as service misuse. Examples include pornography, online gaming, or Internet network file sharing.
13	port	The number (count) of ports on other hosts to which this host has connected, or the number of ports on this host to which other hosts have connected. A port scan is represented by this type of threat.
14	recon	Shows involvement in an attempt to gather information. An IP address with the Feature Attribute of e_s_recon has been seen trying to gather intelligence. A Feature Attribute of e_t_recon suggests that the IP address has been the target of information-gathering attempts.
15	tip	Shows that a service emitted a tip or recommendation that its authors considered useful.
16	virus	Shows involvement in virus communications. An IP address that shows e_s_virus suggests that the host might be spreading viruses. An address that shows e_t_virus suggests that the host might have received virus code.
17	traffic	Records the number of other hosts (incoming and outgoing) that this IP address communicated with. This type of threat is often referred to as a fanout. This kind of behaviour is a “network scan” or “service scan” depending on what was being scanned for (hosts on a network, or hosts offering a given service on the network, respectively). This is in contrast to a “port scan” where multiple ports on a single host are being scanned.

3.3.4.1.2 Vulnerability Instances

Vulnerability instance information will be acquired through the System Services, i.e., SOAP.

3.3.4.1.3 IT Infrastructure

IT infrastructure data pertaining to managed assets (rather than availability data) will be acquired through the native facilities in ISM (bulk loading and processing). This type of data can and should represent two things:

1. “Baseline” or expected assets under administration (and their attributes) – a “configuration management database” (CMDB) of sorts
2. “Discovered” assets resulting from passive and active network discovery

These two data sources will produce four lists: the baseline, the discovered assets, the rogue assets (discovered but not on baseline), and the missing assets (on baseline but not discovered).

3.3.5 Data Storage

Data storage in Intellitactics Security Manager is achieved through a powerful data management schema called the Security Data Warehouse (SDW). Built on industry standard components, such as the Enterprise versions of Red Hat Linux and the MySQL Database Server, each server or node in an SDW deployment contains a three tiered data persistence structure in addition to multi-node SDW architectures that are deployed to meet specific geographic or load-balancing needs. These three tiers include the retention of raw security data in its original unprocessed format (compressed into the RawStore), the post-processing normalized logs (compressed into the ParsedStore), and aggregated and reformatted data into various “fact” tables of the SummaryStore in a relational database. In addition to the event storage, key metadata about the warehouse itself is stored in the relational ObjectRepository. This metadata includes integrity validation information, such as the message digests of all raw log files, timestamp and disk location information, and both top-down and bottom-up linkages between correlated events. Using this elegant highly scalable data warehousing scheme allows for fast query response times, high-speed data insertion, retention of original logs for forensic purposes, and small storage footprints compared to both raw log files and relational databases. GUI-based configuration allows for granular data retention and aging policies for each server in the distributed SDW network, as well as the multi-tier dataflow between SDW nodes (on each ISM server), up to the Central SDW (CSDW).

By default all data acquired by ISM is stored in the SDW, meaning all the original raw data is stored, all the processed data is stored, and aggregates of the data are stored.

There is no plan to export or synchronize any of this data with the JNDMS Data Warehouse. A filtered, correlated, and contextualized subset of the data will be sent to JNDMS for storage and further processing.

3.3.6 Rules / Data Reduction

The ISM rules engine is principally designed to process events through rules, as well as present data through visualizations and data tables. These two functions require some additional artefacts within the working memory of the rules engine itself.

The ISM rules as designed for JNDMS are of the following categories:

- **Input Rules:** responsible for acquiring data (principally security events), creating, and storing events.
- **Filter/ Routing Rules:** remove events from the real-time stream or direct events to the appropriate internal or external entity (rule, subsystem, etc.).
- **Add Context Rules:** tag events with additional information based on knowledge bases (vendor provided context, environmental context, temporal context). These add new attributes and values to events as they flow through the system.
- **Correlation Rules:** perform correlation with events. Correlation can take many forms, some examples include: time based correlation, correlation on a field or attribute, rate based correlations, and threshold based correlation.
- **Alert Rules:** create and issue Alerts. These are also responsible for transforming the data into a format suitable for native consumption by JNDMS.
- **Control Rules:** process system control events. These meta-rules control the behaviour of ISM in general and the rule engine in particular. This also includes handling of timing events.

The other significant artefacts in the rule engine are:

- **State:** dynamic information (objects and associated attributes) stored in memory. It is important to note that rules themselves are just state and can be manipulated by other rules.
- **Tables:** information stored in hash tables for rapid access – typically this type of information is fairly static and is used by rules as a “knowledge base” for a particular type of lookup.
- **System objects:** configuration objects that deal with the ISM system itself, rather than the security event data being processed.

3.3.6.1 Correlation Objectives

The goal of correlation and analysis on security event data is to answer questions relevant to SA for CNL. By designing our correlation rules to answer a set of questions, we increase the chance of producing relevant correlations. This section enumerates the types of manipulations we would like to do on the data before it is passed to JNDMS. Pertinent evidence is required to answer each of these questions provided by the data source (such as an IDS or log file), environmental context (such as details of the assets involved), and temporal factors (such as watchlists or event rate information calculated by the rule engine over time). This evidence is required to answer the following questions:

- Who are the attackers? (IP, DNS, rDNS, whois, PoF fingerprinting, etc.)
- What is being attacked?
- What is the sophistication of the attack? (multiple signatures vs. known worm)
- Are the attackers known offenders? (watchlist)
- Are the attacked hosts on a watchlist?
- Are the user IDs involved on a watchlist?
- Internal to Internal? Internal to External? External to Internal?
- Sources relationship to targets?
- What's happened recently?
- What kind of event is it? (taxonomy, event type)
- What needs to be dealt with first? (priority)
- Are multiple simultaneous events related or separate?
- Are multiple vulnerability instances related or separate?
- What bad things this host has done? What bad things been done to this host?
 - Is the target exposed to several different signatures? (event types and counts)
 - Is the source common among many signatures? (event types and counts)
- Is the event type rare (and not usually seen in that area of the network) or do we see it there all the time?
- Where is this host on the network?
- What are the set of assets that are impacted?
- What is the vulnerability status of the hosts involved? (CVE match)
- What are the operating systems of the hosts involved? (CMDB or fingerprinting)
- What ports are open on the set of hosts involved? (match to t_port of attack)

- What missions are impacted? (or campus locations)
- What is the patch status of the set of hosts involved? (or last patch date)
- Who manages the set of hosts involved?
- Where is the location of the hardware? (Geographical Information System [GIS], or more likely deployed versus not-deployed)
- Have numerous high priority events been witnessed?
- Has the source IP touched many targets? (breadth)
- Has the source IP touched more significant / critical targets? (focus)
- Has a target become a source of an event type? (infection)
- Have the event types been classified as dangerous? (exploit versus recon, etc.)
- Does the event represent a policy violation? (telnet in a DMZ)
- Has the rate of events of a certain type exceeded a baseline threshold?
- Is there an unusual amount of usual traffic? (too high or too low)
- New “event type” seen in zone?
- New user_id seen in zone?
- User_id very frequent? (with whitelists)
- User_id not seen in long time? (with whitelists)
- Time based watchlists?
 - “Watch” probe/ recon sources on list for 2 weeks?
 - “Watch” compromise attempt sources for 6 months?

During cycle 3 of the JNDMS development the trade off between providing these correlations in the SIM as opposed to within JNDMS. During this development cycle the correlation was generally done within the JNDMS to prevent data loss and ease implementation.

3.3.7 ISM Alarm Escalations to JNDMS

The objective of this document is to describe the mechanisms by which network security events are repressed or escalated to JNDMS as Incidents. There are several places where the JNDMS/ security administrator can make adjustments to the default settings to change various thresholds and filters to this escalation process. Understanding the escalation process and the ways it can be modified will allow the JNDMS administrator to tailor it to their environment and needs. The escalation process described uses the example of the JNDMS deployment on the DREnet as the context.

3.3.7.1 Overview

Security events in JNDMS go through several steps before they become an incident. The steps are as follows:

1. Malicious or anomalous traffic is detected on the network by a Network Intrusion Detection System (NIDS) sensor, raising an alarm
2. The alarm is sent from the sensor to the NIDS database
3. The Security Information Management (SIM) data acquisition (DA) server periodically polls the NIDS database for new alarms
4. The SIM rules and correlation servers take alarms from the DA and prioritize them using a “Risk Scoring” algorithm
5. The SIM Security Operations Center (SOC) server takes the prioritized set of alarms and slots them into buckets (such as “Correlated Alerts”, “Primary Alerts”, “Secondary Alerts”, and so on) based on user-defined thresholds
6. The JNDMS alarm escalation process periodically takes the alarms (and their associated events) from the highest priority buckets (Primary and Correlated Alerts) and sends them to the JSS/DSS (Decision Support System)
7. The DSS uses a set of user-defined criteria to determine which types of alarms should become JNDMS Incidents and which become JNDMS “events”
8. The DSS compares the new alarms with the current set of Incidents and represses duplicates

Each of these steps is described in greater detail below, along with guidance on how to modify the escalation process (where applicable).

3.3.7.2 NIDS Sensor Detection

The Network Intrusion Detection System (NIDS) in this deployment is the open source tool Snort. Each Snort sensor monitors the flow of network traffic at the perimeter of the various security zones on the DREnet.

Each sensor raises alarms based on a set of pattern matching rules called signatures. Signatures are automatically updated on a daily basis to keep up to date with the latest threats.

3.3.7.2.1 User configurable parameters

A user can control which alarms are generated at this level by modifying the signatures running on the sensors.

Signatures can be added or removed as required. Signatures that generate a large number of false positives on a given sensor are typically removed. This can vary depending on environmental factors of the monitored network.

Signatures that are generally useful, but have a known false trigger can be modified to ignore that specific trigger, for example a given host that does network vulnerability scans might be explicitly ignored by a sensor so as not to generate thousands of false attack alarms when it runs

The signature modifications process is normally performed by the IDS system administrator on an "as needed basis". The DREnet Management contractor performs this task on behalf of DRDKIM.

3.3.7.3 NIDS Database

The DREnet Snort infrastructure uses Barnyard to store alarms from all DREnet Snort sensors in a MySQL database. Barnyard is an open source tool designed to work with Snort to take alarms from multiple sensors and store both the alarm (and in many cases the payload of the packet that triggered the alarm) into a database in an efficient manner.

This step in the data chain between sensor and JNDMS has no user configurable parameters.

3.3.7.4 SIM Data Acquisition

The DREnet corporate SIM (Intellitactics Security Manager – ISM) uses a “Data Acquisition” server (DA) to periodically poll the DREnet Snort database for new alarms and pull them into the rules system for processing. A custom rule has been written on the DREnet ISM to create two copies of those alarms. A cron job running on the DREnet corporate ISM moves the copy of the harvested alarm data over to the JNDMS integration server on the DREnet management LAN. A service on the JNDMS integration server sends the alarms to the “inbox” of the ISM DA on the NIO SOC LAN. Another cron job ensures that if data delivery is not possible, files older than a certain threshold are deleted, so that if the ISM DA is unavailable for a long time, it does not fill the drive on the JNDMS integration server, or overwhelm the DA once it comes back online.

3.3.7.4.1 User configurable parameters

The user of the DREnet corporate ISM can control the polling interval of the DA to the Barnyard DB, which affects alarm latency and performance of the servers.

The user can control how long alarms are cached on the JNDMS integration server before they are discarded, by modifying the data delivery cron job.

3.3.7.5 SIM Rules and Risk Scoring

The ISM “Threat Detector” (TD) and “Threat Evaluator” (TE) use a system of correlation, aggregation, escalation rules, and filters to group and prioritize alarms.

One of the factors used to prioritize alarms in ISM is the operational value of the assets involved. This asset value, called “Operational Risk” in the ISM context is expressed as a number in the JNDMS. The DSSAsset class in JSS has a simAssetValue member that corresponds to the ASSET.IMPORTANCE column in the database. The range of this value is **0 to 10**. This new valid range matches the asset value range.

NOTE: The value range from 1 to 5 that was specified in the original design is no longer valid.

See the JSS SIMImpactReport class:

```
com.mdacorporation.jndms.JSS.Core.publish.SIMImpactReport
```

SIMImpact Report sends a list of assets to the SIM. Only assets with an ASSET.TYPE of 'primary hardware' and ASSET.IMPORTANCE value greater than zero are included in the list. These assets are retrieved using a named query ('jss.publish.report.sim') that is stored in the orm.xml file in the JSS project.

The creation of SIMImpactReports is configured through the PublishProfileCreator class:

```
com.mdacorporation.jndms.JSS.Core.publish.PublishProfileCreator
```

The PublishProfileCreator is instantiated in the JSS GlobalEventQueue class constructor:

```
com.mdacorporation.jndms.JSS.Core.GlobalEventQueue
```

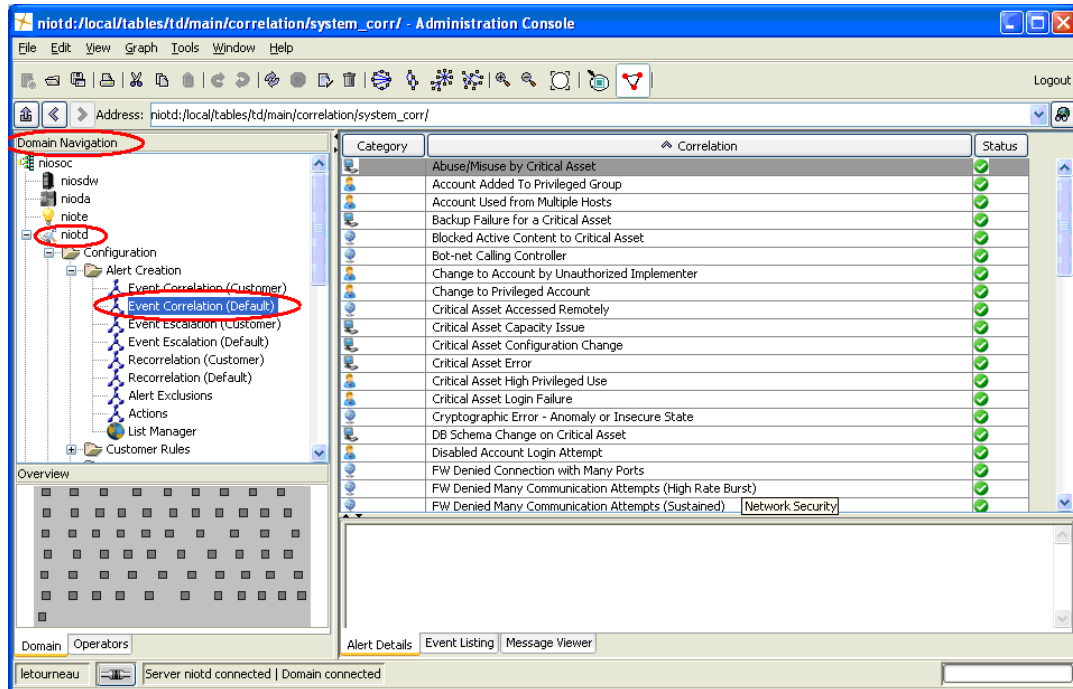
3.3.7.5.1 User configurable parameters

There are a number of user configurable parameters for this step.

Correlation Rules

ISM has a number of user modifiable correlation rules which come with the product, and others can be created at the users discretion. These rules can be enabled/ disabled, or modified from within the ISM Administration Console. While there are a number of steps in the data flow within ISM which precede the correlation rules, the first place relevant to alarm escalation to JNDMS is in the “Threat Detector” (TD) subsystem. The correlation rules are applied to security events as they pass through the TD, weighing each event against such criteria as environmental data, source and target history, vulnerability status, similar events which have occurred in a specified time period, and so on. Many of the correlation rules generate a new alarm of a “correlated” type rather than passing the original event on through the escalation path; a good example of this is when many events are taken together to represent a single correlated alarm.

To view and modify the default correlation rules, log in to ISM using the Administration Console and navigate to the TD in the “Domain Navigation” panel. A set of nested folders under this subsystem (shown following as the niotd) contain bookmarks to various key settings and configuration panels in ISM.



Modifying Correlation Rules

Each of the correlation rules in the TD have one or more user modifiable parameters, which are accessed by opening the “meta panel” specific to the rule. A meta panel is an ISM dialog box with user editable controls specific to the system element being modified and is typically accessed by double clicking on an item (such as a rule) or using a bookmark in the Domain Navigation panel. An explanation of the correlation rule, along with the modifiable fields is found in the meta panel.

This example shows a rule detecting “Abuse/ Misuse by a Critical Asset” and allows the user to define what “operational risk” (asset value) is used as the threshold of “critical”. Changing the operational risk threshold (say from the default of 2 up to 4) will limit the events that this rule will consider to those originating from assets with an asset value of 4 or greater (instead of > 2).

DN0678
Issue 4/2: 06 February 2012

The screenshot shows the Administration Console interface for configuring an alert. The main window is titled "Abuse/Misuse by Critical Asset" and has tabs for "Attributes" and "Advanced". The "General" section contains the following fields:

- Label:** Abuse/Misuse by Critical Asset
- Description:** This alert triggers when a critical asset is seen as the source of abuse/misuse activity.
- Comment:** This operator selects events where the source's operational risk is beyond a threshold and the taxonomy type represents misuse/abuse activity. You may change the operational risk threshold representing critical hosts by changing the value listed in the Inbound Filter configuration section.
- Inbound Filter:** Match (Edit button), Outbound Alert, Alert Keys (Edit button), Action: default

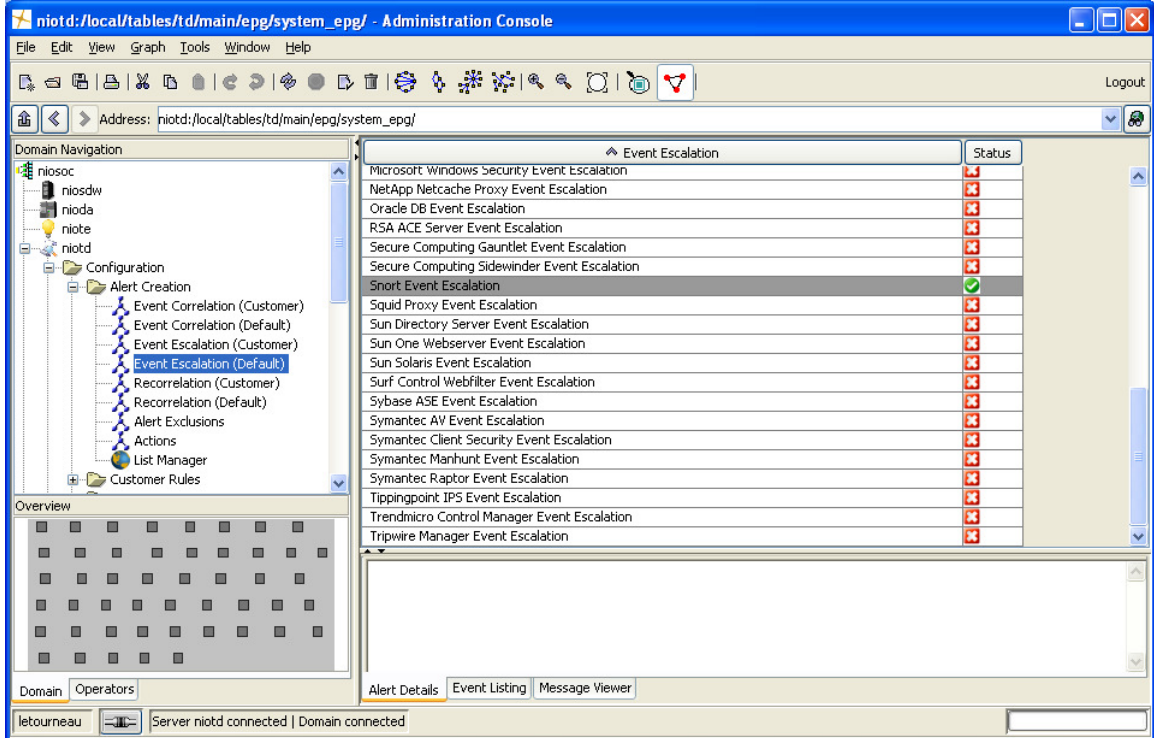
The "Table Properties" dialog box is open, showing a table with the following data:

Key	Function	Value	And/Or
s_operational_risk	greater_than	2	AND
nsm_type	like	{ids}.detect[fw],aut...	AND

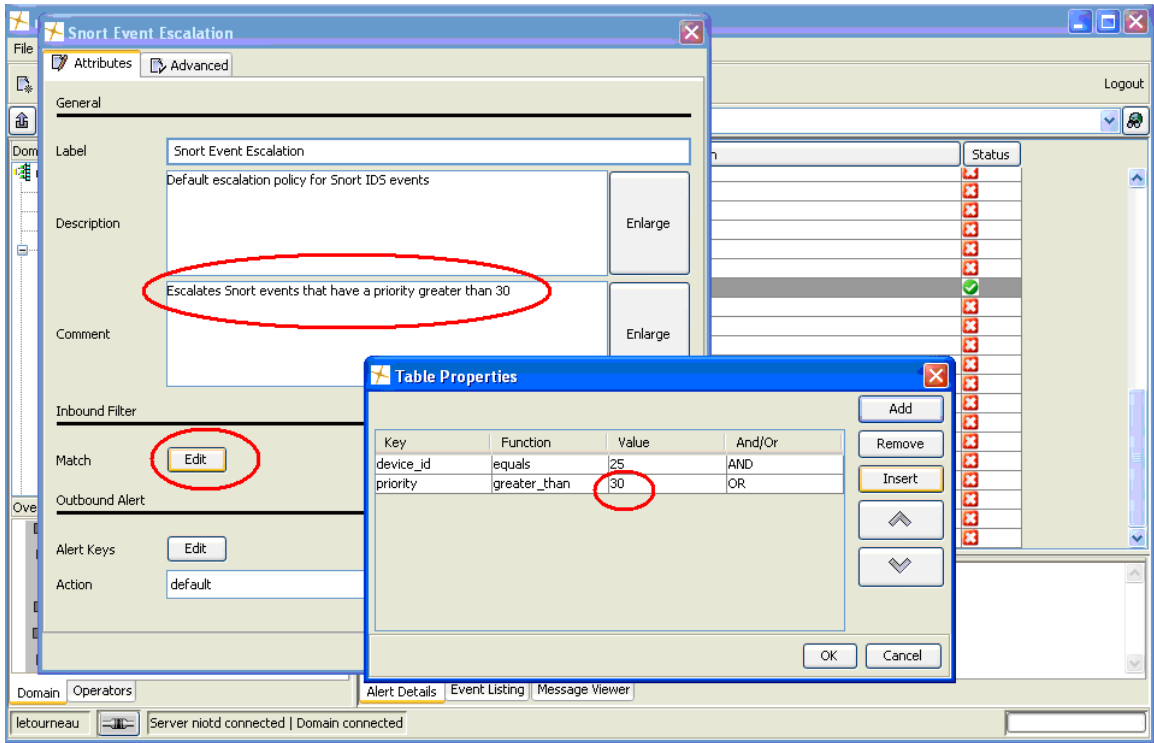
Red circles highlight the "s_operational_risk" key in the description, the "Match" button, and the "2" value in the table. The status bar at the bottom shows "Server niotd connected | Domain connected".

Escalation Rules

After an event is processed through the ISM correlation rules, the TD also has a set of “escalation rules” which allow events to be assessed against a set of criteria which determines if they are significant enough to escalate for further processing as a high priority alarm.

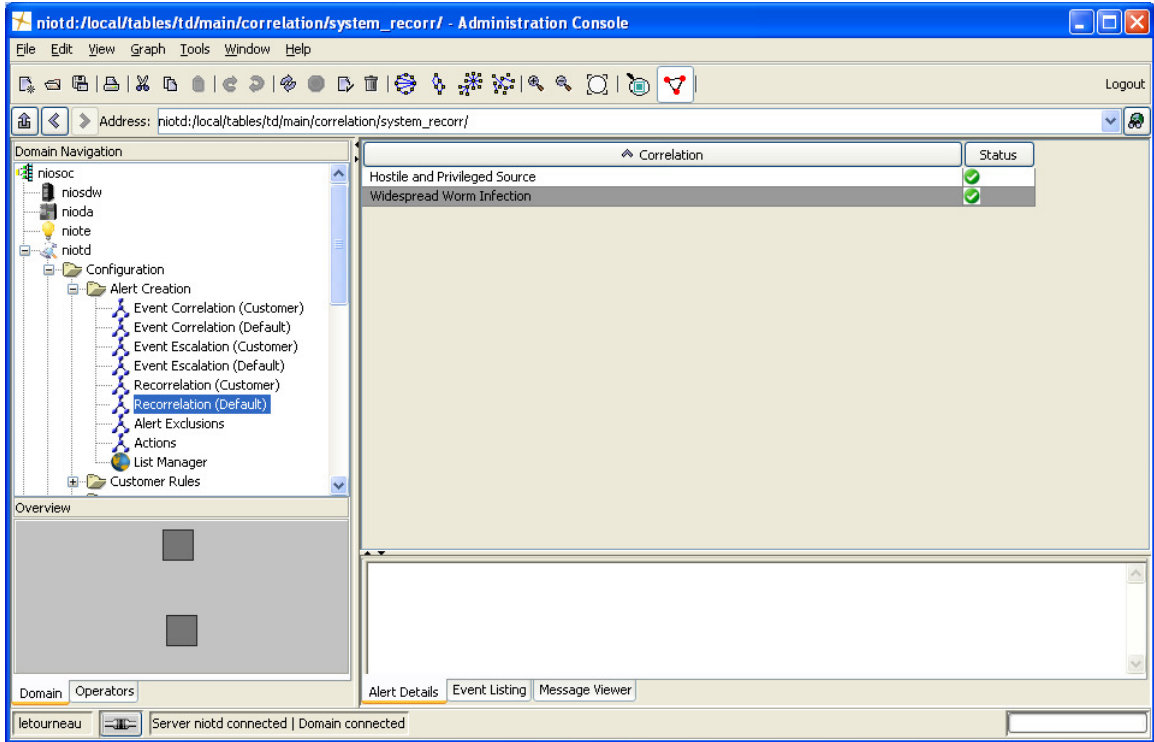


These escalations are also modifiable via a meta panel in a fashion similar to the correlation rules. This example shows that the user can modify the escalation threshold for Snort alarm priority.

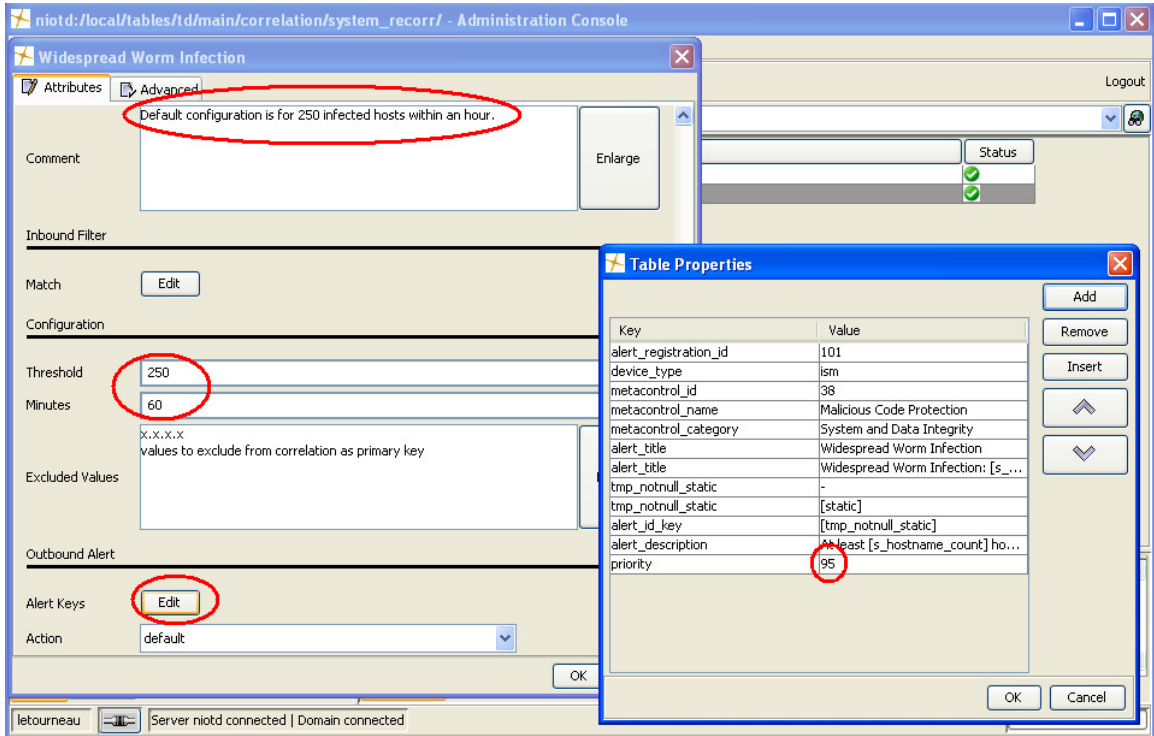


Recorrelation rules

The ISM TD is also responsible for finding patterns (correlations) amongst the other high priority alarms (escalations) and correlated alarms previously seen by the system. Each correlated alarm is fed through “recorrelation rules” which weigh them in the context of the other alarms to determine if there is a larger pattern of malicious behaviour at work, such as a worm outbreak.



The recorrelation rules are modifiable in the same fashion as other TD rules by using the meta panel. This example shows the rule used to detect worm-like behaviour and how the user can modify such parameters as the number of events, time window, and the priority of the generated correlated alarm.



Risk Score

A key part of the ISM rules system is a “risk scoring” algorithm for each alert (correlated or individual). This risk score takes into account a number of historical and environmental factors in making the determination as to how important it is to the ISM user. In the ISM “Security Operations Center” (SOC) subsystem, the user can select any alarm and view the factors and their score contribution which determine the ISM risk score.

These factors are:

Alert priority: a measure of how important this alarm type is relative only to other alarm types

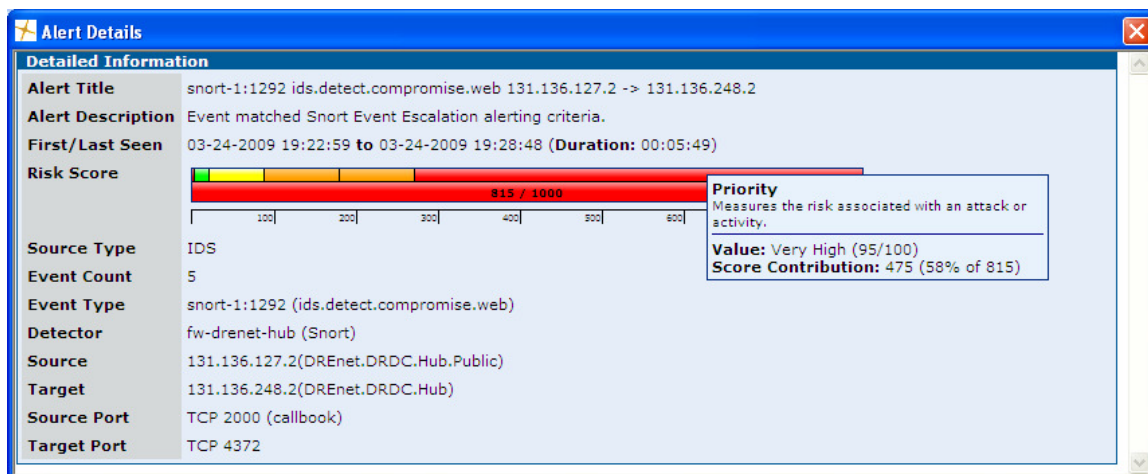
Source and Target vulnerability: determined by using the quantity and severity of vulnerabilities on the source and target of the event (if known)

Source and Target history: determined by assessing the history of events (known to the SIM) on both the source and target

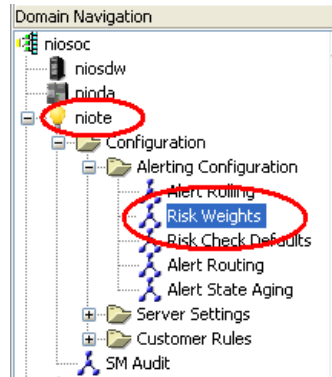
Source and Target Compliance: a measure of how important the source and target systems are from a regulatory compliance perspective

Source and Target Operational value: a measure of how important the source and target systems are in terms of operational value (asset value)

The risk score is normalized to produce a value between 1 and 1000.

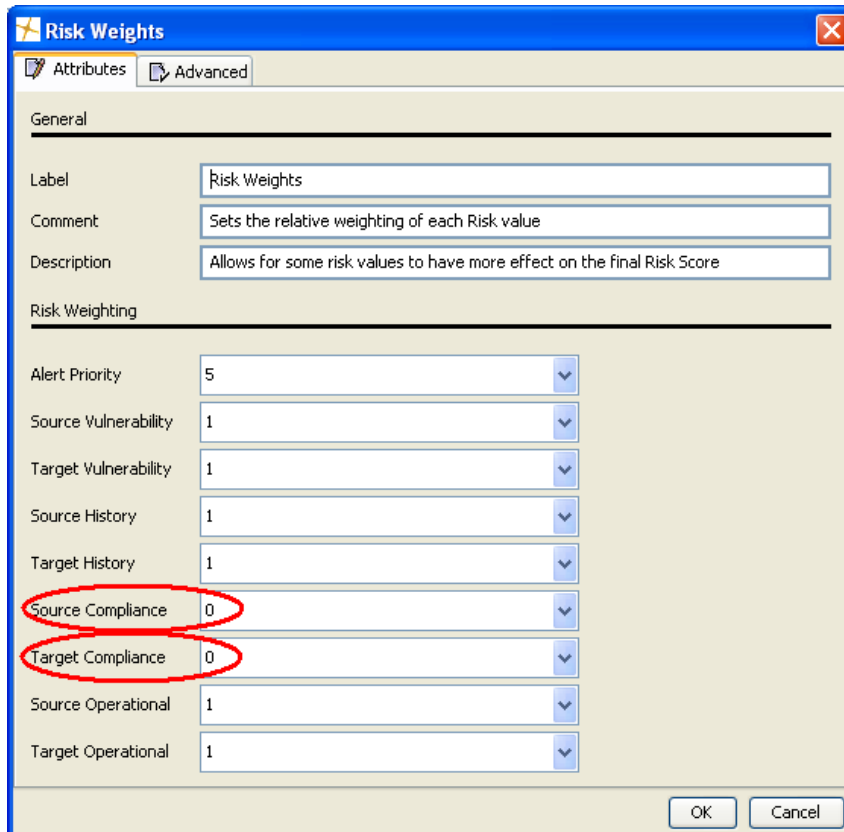


Users can modify the weight each of these factors has in the Risk Scoring formula; the meta panel is accessible through the “Risk Weights” bookmark on the ISM “Threat Evaluator” (TE) subsystem.



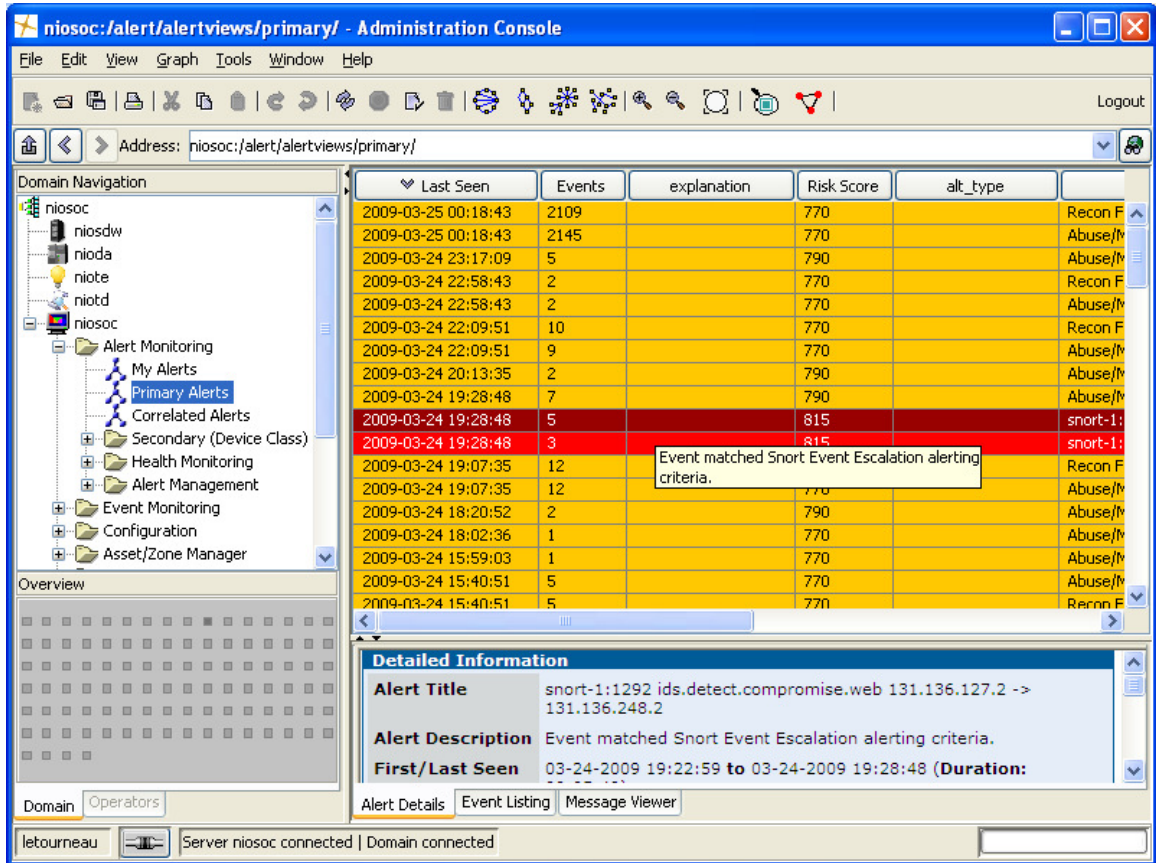
Each factor that contributes to the overall risk score can be assigned a weight value of between 0 and 10, which allows the user to bias the risk score toward those factors they consider most important in their environment. Each “weight” number becomes a multiplier for that factor in the overall risk score, which is then normalized to always produce a value between 0 and 1000.

This example in the DREnet context the assets have not been assigned a “regulatory compliance value”, so these factors have been turned off in the Risk Scoring formula.



3.3.7.6 SIM Presentation Layer

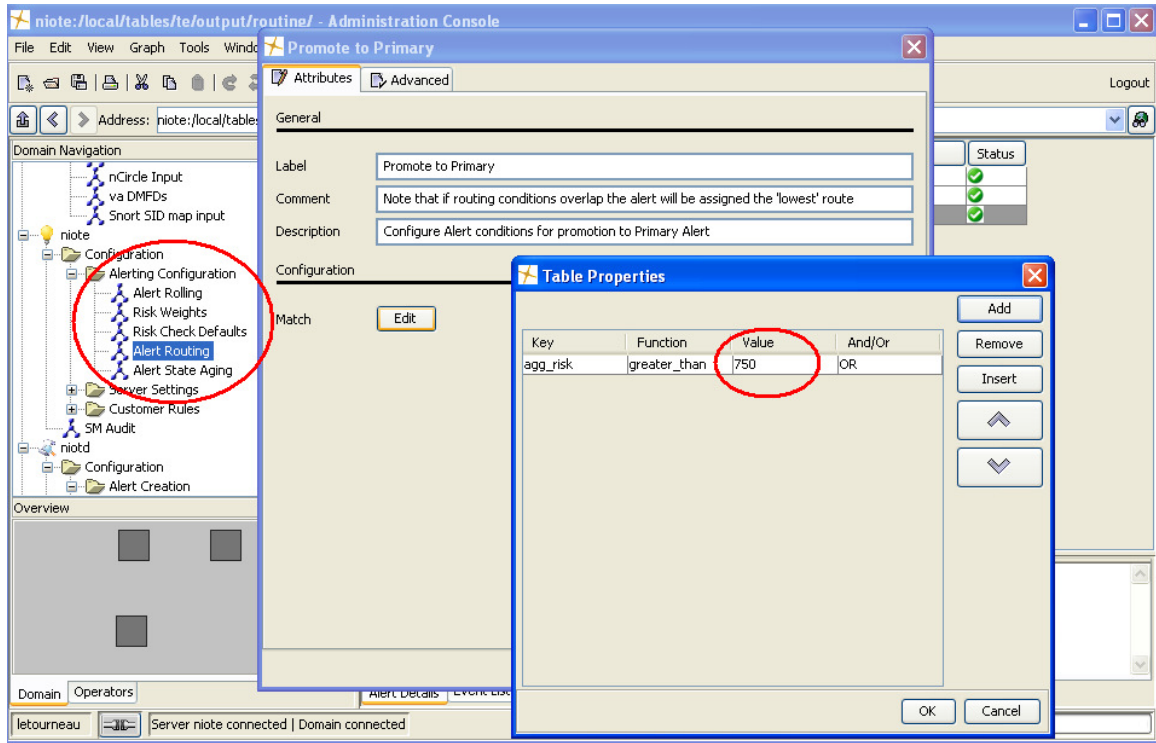
At the level of the ISM SOC, alarms that have a risk score exceeding a user defined threshold are presented to the user in the “Primary Alerts” view. These include correlated, escalated, or re-correlated alerts.



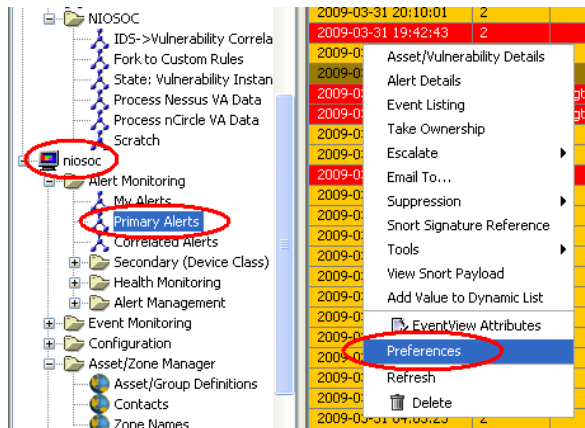
3.3.7.6.1 User configurable parameters

The user can control what Risk Score threshold is used as the cut-off for inclusion in the Primary Alerts view. This also determines which alarms are escalated to JNDMS in the context of the DREnet deployment.

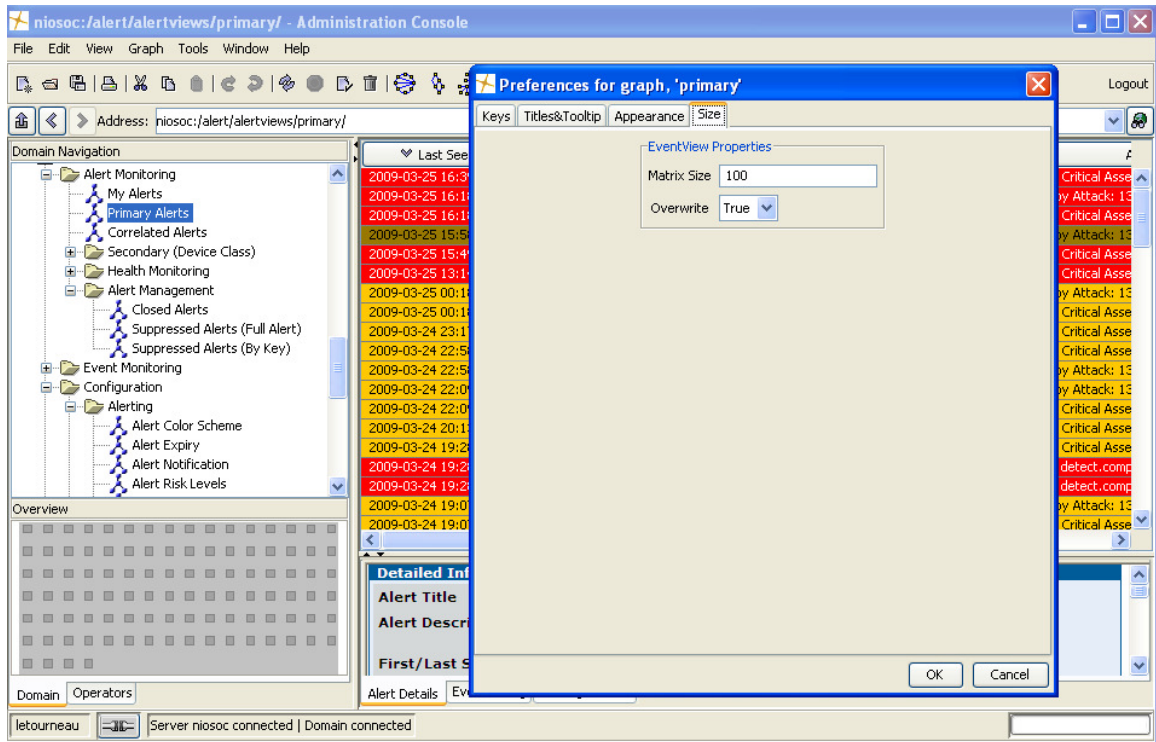
In this example we see that on the TE, the user can change the Primary Alerts threshold in a meta panel, accessed through the “Alert Routing” bookmark.



Additionally, the user can control how many alarms are held in the Primary Alerts view. This is accessed from the ISM SOC under the Primary Alerts view by right clicking on any alarm in the view and selecting “Preferences”.

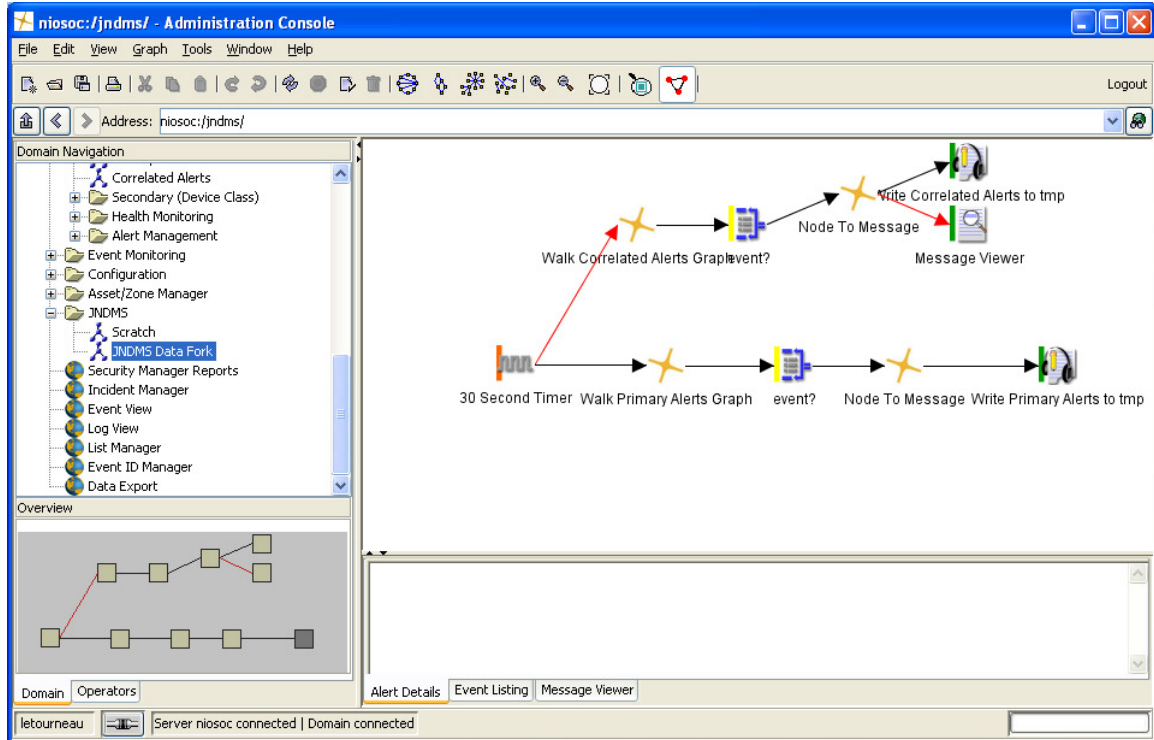


The current setting in the JNDMS ISM retains up to 100 Primary Alerts at a given time. As the hundred and first (101st) event comes in, it will overwrite the oldest alert in the view, and so on.



3.3.7.7 Escalate SIM alarms to JNDMS

A timer process on the ISM SOC server iterates through the alarms in the “Primary Alerts” graph every 30 seconds and writes a copy of the alarms to a temporary directory.



A system daemon (/etc/init.d/jndms_sendalerts) watches the temporary directory and sends the alarms to the JSS. A data cleanup cron job periodically runs to clean out the temporary directory of all files older than 1 hour (to ensure that there is never an overwhelming backlog of events waiting for escalation to the JSS if it is offline for some time).

3.3.7.7.1 User configurable parameters

The data retention period and frequency of the data cleanup script can be modified as required.

3.3.7.8 Categorize alarms as Incidents or Events

The DSS maps the SIM correlated alert and alarm types to JNDMS Incident Type using a pair user-definable properties files. These type mappings determine the JNDMS Incident type, and whether or not the alarm in question is an “event” (INCIDENT.INCIDENT=N) or an incident (INCIDENT.INCIDENT=Y) which is presented in the JNDMS portal.

3.3.7.8.1 User configurable parameters

There are two files that are user configurable at this step. The files are found in JSS\WebContent\WEB-INF and specify the mappings for correlated or single (high priority) alarms respectively. It is also worth noting that the JNDMS Incident Types themselves could potentially be modified, but this would require more research to understand the impact that those changes might have on the system.

The following file is used to map ISM “correlated alert” types (e.g. “Possible DDOS Target”) to JNDMS Incident types (e.g. “Denial of service”). It is also used to specify if the alert type is considered an “Incident” in JNDMS terms or not (set by the true/false field). If a new ISM correlated alert type is passed to JNDMS for which a mapping does not exist, it is noted in the JSS log file on the JNDMS Portal system. If the file is modified, the JSS must be restarted for the changes to take effect.

correlation-properties.xml

```
<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE properties SYSTEM "http://java.sun.com/dtd/properties.dtd">

<properties>

<entry key="IDS Many Alerts Targeting Critical Asset">Malicious logic - worm, true,
TARGET, 1.0, 0.05, 0.2</entry>

<entry key="Possible DDOS Target">Denial of service, true, TARGET, 1.0, 0.05,
0.2</entry>

<entry key="Recon Followed by Attack">Reconnaissance, true, SOURCE, 1.0, 0.2,
0.05</entry>

<entry key="Restricted Source Port">Reconnaissance, false, NONE, 0.1, 0.1, 0.0</entry>

<entry key="Restricted Target">Reconnaissance, false, TARGET, 0.1, 0.3, 0.1</entry>

<entry key="Restricted Target Port">Reconnaissance, false, NONE, 0.1, 0.3, 0.1</entry>

<entry key="Abuse/Misuse by Critical Asset">Reconnaissance, true, SOURCE, 1.0, 0.2,
0.1</entry>

<entry key="Critical Asset Accessed Remotely">Reconnaissance, true, TARGET, 1.0, 0.1,
0.3</entry>

<entry key="IDS Alert to Vulnerability Match">Compromise, true, SOURCE, 1.0, 0.8,
0.8</entry>

<!-- Remove the following when corrected -->

<entry key="snort-1">Policy violation - other, true, NONE, 1.0, 0.8, 0.8</entry>

</properties>
```

The following file is used to map the ISM individual alarm “taxonomy types” to JNDMS Incident types. To accommodate the thousands of taxonomy types, the first three levels of the (hierarchical) taxonomy name for the alarm are used to match many related alarm types to the appropriate JNDMS Incident type (e.g. everything in the “ids.detect.exploit.*” hierarchy is mapped to “Malicious logic – other”). If the file is modified, the JSS must be restarted for the changes to take effect.

nsm-properties.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE properties SYSTEM "http://java.sun.com/dtd/properties.dtd">
<properties>
<comment>NSM TYPE</comment>
<entry key="ids.detect.agent">Malicious logic - trojan horse</entry>
<entry key="ids.detect.anomaly">Malicious logic - other</entry>
<entry key="ids.detect.auth">Reconnaissance</entry>
<entry key="ids.detect.compromise">Compromise</entry>
<entry key="ids.detect.conf">Denial of service</entry>
<entry key="ids.detect.corrupt">Denial of service</entry>
<entry key="ids.detect.deliver">Malicious logic - worm</entry>
<entry key="ids.detect.disclose">Reconnaissance</entry>
<entry key="ids.detect.dos">Denial of service</entry>
<entry key="ids.detect.error">Policy violation - misconfiguration</entry>
<entry key="ids.detect.evade">Reconnaissance</entry>
<entry key="ids.detect.exploit">Malicious logic - other</entry>
<entry key="ids.detect.fw">Reconnaissance</entry>
<entry key="ids.detect.insecure">Policy violation - misconfiguration</entry>
<entry key="ids.detect.misuse">Policy violation - unauthorized use</entry>
<entry key="ids.detect.nocompromise">Reconnaissance</entry>
<entry key="ids.detect.recon">Reconnaissance</entry>
<entry key="ids.detect.request">Reconnaissance</entry>
<entry key="ids.detect.spoof">Reconnaissance</entry>
<entry key="ids.detect.svc">Reconnaissance</entry>
<entry key="ids.detect.throttle">Reconnaissance</entry>
</properties>
```

3.3.7.9 Repress Duplicate Incidents

The event count of how many security events a JNDMS Incident represents is available in the ISM Primary Alerts view, however this information is not held within JNDMS. After categorization the DSS refers to the Incident table; any alarm which represents an additional alarm with the same attributes and values as one that already exists in the Incident table is discarded, rather than creating a duplicate entry. For example to avoid multiple duplicate Incidents corresponding to a given correlated “port scan” alarm, each subsequent event belonging to the same port scan Incident is discarded by JNDMS (but still available in ISM).

There are no user configurable settings for this step.

3.3.8 Scenarios

The SIM component is the “information security” bridge between the CND environment and the JNDMS core. In the CONOPS it will still be a starting point for investigating security incidents and, as such, will retain all security data, have tools for incident response, security visualization, and so on. Its primary in-scope function in JNDMS is to acquire security events from the CND environment and forward the relevant subset of that information to the JNDMS core.

3.3.8.1 Example Scenario

A JNDMS is deployed on an Internet connected network for the purposes of providing SA of that CND environment. ISM is used to acquire security data from the CND environment in the following manner:

Cisco Secure IDS sensors are deployed at various chokepoints on the Wide Area Network (WAN), including behind the firewalls to the redundant Internet links. ISM “subscribes” to data feeds directly from the sensors using Remote Data Exchange Protocol (RDEP) – an SSL encrypted XML based data feed for alert data. RDEP does not automatically push new alerts as they occur, so ISM polls each sensor every 30 seconds and asks for “all events newer than timestamp X”. The timestamp cursor and other RDEP details are held in state for each sensor by ISM. New IDS alerts are written to the SDW and processed by the ISM rules.

PIX firewalls are deployed at the Internet perimeter and internally to define various security zones on the network. Each is configured to deliver its logs to ISM using standard syslog. Data streams into ISM in real-time from each PIX where it is aggregated and processed in 60-second time windows for performance reasons.

Windows event logs from the NT Domain controllers are harvested by a “Windows Event Monitor” (WEM) agent sitting on a server dedicated to that purpose. Since Windows has no facility for remote delivery of system logs, software that talks to the Windows API is needed to trap the events as they occur. WEM runs on a dedicated server that pulls the logs from each Domain Controller and then formats them into the ISM native protocol (LMP) for delivery to a downstream ISM server.

After data acquisition, ISM processes each kind of alert by storing it in the SDW and sending either aggregated or atomic (single event) data through the rules engine. Those events above the threshold of importance are forwarded on to JNDMS through the web services.

3.4 Data Transformation

3.4.1 Overview

The data transformation component in JNDMS is a set of services that can be used by agents or components to perform data transformation. This component does not represent a physical entity that data is routed through, but rather a set of tools, APIs and interfaces that aid in the development of other components.

3.4.2 COTS Selection

3.4.2.1 CA Advantage Data Transformer

One of the chosen products for this function is CA Advantage Data Transformer (ADT). This CA product supports a wide range of data transfer and transformation features. A high level summary is provided here:

The Graphical Mapper and workflow manager deliver finely controlled data transformation and data integration applications, including complex data calculations and transformations, while simplifying database connectivity and data transformation. You can access, consolidate and transform all types of internal and external structured data into information assets. Near real-time data synchronization and replication across your diverse data platforms is also supported. Automatic data type conversions reduce the development effort required to move data between disparate sources and targets. A built-in interface for database, file and groupware systems eliminates the need for you to learn database-specific APIs, reducing development and maintenance time and allowing your developers to concentrate on the transformation process.

ADT supports multiple concurrent developers and provides advanced multi-tasking capabilities. Reliable performance is ensured by error handling (control, recovery and notification) capabilities, while an interactive debugger assists in quick production and testing of effective data movement scripts.

Reuse and standardization are promoted through centrally stored and accessed user-defined functions that are callable by transformation scripts. Network independence facilitates the use of all common networking protocols and architectures with source and target databases residing on any platform. A scalable architecture provides the backbone necessary for enterprise-wide applications. You can initiate data transformation, manipulation and integration manually through the built-in scheduler, by an external scheduler, or based on another data movement request or outside event, all managed by its graphical workflow manager.

The CA Advantage Data Transformer was not used in the final DREnet deployment.

3.4.2.2 XML Translations

Much of the core of the JNDMS is based on XML, which provides its own data translation services. These translation services are available in a number of languages; however Java is the language most prevalent within JNDMS.

The Java language defines the Java API for XML processing (JAXP, <http://java.sun.com/webservices/jaxp/>). These interfaces bring core processing capabilities directly into the language. One of the key technologies for XML to XML translations is the XML Stylesheet Language Transformations (XSLT), which allows transformations to be described in a stylesheet and common tools, such as the JAXP, can be used to perform the translation.

3.4.2.3 Custom Translations

There are some translations that are more complex or require special attention and these translations are generally done through custom translations in Java. One case of a common translation used is to translate event mechanisms in various tools or components into the appropriate calls for JNDMS. A command line tool was created to allow for a number of these translations to occur (see 3.4.3, JSS Client).

3.4.3 JSS Client

A custom client can be used to translate command line parameters into a SOAP call for common events within JNDMS. This client is a Java application that adheres to the SOAP / WSDL interface of the JNDMS System Services (JSS).

Table 14 lists the available operations.

Table 14: JSS Client Operations

Operation	Parameters	Notes
SIM	Type	Classifies type of event (see Table 13: Security Feature Categories)
	Sub type	More detailed classification of event, if available (see Table 17)
	Source ip	IP address of the source of the event
	Target ip	IP address of the target of the event
	Sensor ip	IP address of the sensor recording the event
	Sensor event ID	An identifier that can be used by the referenced sensor to identify this event. This identifier is likely meaningless outside of the context of the sensor.
	Sensor CVE ID	One or more vulnerability references relevant to this event.
	Base Priority	The SIM's recommendation for the priority for this type of event.
	SIM Priority	The SIM's recommendation for the priority of this event after correlation.
	Correlation CVE	Additional vulnerability references.
	Sensor Time	Time of this event as recorded by the sensor.
EIM	Type	Type includes ip_up, ip_down, ip_change, new_interface, link_info and alarm.
	Source	Identifier for which component is reporting.
	Source_inst	Used in combination with the source field to uniquely identify the source product and particular reporting agent or instance.
	Zone_id	This is used to identify which network zone the agent is reporting from. This is used to ensure there are no

DN0678
Issue 4/2: 06 February 2012

Operation	Parameters	Notes
		ambiguities in cases where network address translation is being used.
	IP Address	IP address for event. This will refer to the source or current IP addresses for event types that require multiple addresses, such as source and destination.
	Name	This is the name of the entity being reported on and depends on the type of event. It will refer to the source or current address and generally will be the host name.
	Label	This is a label applied to the entity being reported on. It refers to the same entity as the ip address and name fields.
	UUID	This is an identifier for this entity (same as ip address and name fields) that is unique for this source.
	Class	The field depends on the source that is reporting and may be used to remove ambiguity in reports.
	Create Date	Date and time of event.
	Alt IP	This is an alternate IP address used by some event types. It will refer to the destination or previous address related to this event.
	Alt Class	This is the class of the entity (source dependant) referenced by the alt ip, alt name and alt uuid fields.
	Alt UUID	This is the unique identifier (source dependant) for the entity referenced by the alt ip, alt name and alt uuid fields (some or all may be present).
	Alt Name	This is the name for the alternate entity.
	Alt Label	This is the label given to the alternate entity.
	Severity	This allows the reporting agent to identify its assessment of the severity of the event.
	Status	This allows the reporting agent to identify its assessment of the status of the events, especially when reporting alarms.
CME	Source	An identifier for the source of this information.
	URL	The URL to the XML data. The

DN0678
Issue 4/2: 06 February 2012

Operation	Parameters	Notes
		format must meet the schema for the CME (http://cme.mitre.org/).
CVE	Source	An identifier for the source of this information.
	URL	The URL to the XML data. The format must meet the schema for the CVE (http://nvd.nist.gov/).
Safeguards	Source	An identifier for the source of this information.
	URL	The URL to the safeguard data. (see Sample Safeguard Data Inputs).
Operations	Source	An identifier for the source of this information.
	URL	The URL to the operations data. (see Sample Operations Data).
Assets	Source	An identifier for the source of this information.
	URL	The URL to the asset data. This data must be formatted according to the source.
Dp_update	Source	An identifier for the source of this information.
Dp_enable	Source	An identifier for the source of this information.
Dp_disable	Source	An identifier for the source of this information.
V_scan	Source	An identifier for the source of this information.
	URL	The URL to the XML formatted vulnerability scan. (see Sample Vulnerability Report).
Cap (Common Alerting Protocol)	Source	Identify the source of the event.
	URL	The URL to the XML event as defined by the Oasis (http://www.oasis-open.org/home) schema.
Jndms_xml	Source	Identify the source of the information.
	URL	The URL to the XML event as defined by the JNDMS reporting schema. This event type can be used to report on many JNDMS relationships including assets, operations, services and the dependencies between them.
Scan_dir	Source	Identify the source of the information.
	Dir	Directory to scan for incoming events (used in the sharing scenarios)

DN0678
Issue 4/2: 06 February 2012

Operation	Parameters	Notes
	interval	The time interface to wait before checking the directory for new events.
View	Ids	A list of event IDs to view.
	Type	'summary' or 'all' identifies what information is proved for each event.
Remove	Ids	This will remove the given ID from the queue.
[all]	saveEvent	Any event can be saved for later play back. This parameter is set to true (saveEvent=true) or false.
	Dir	If events are stored, this is the directory they will be stored in.

3.5 JNDMS Data Warehouse

A Data Warehouse is a repository of information accumulated from a variety of sources. Prior to being stored in the JNDMS Data Warehouse, this multi-source data will be transformed then aggregated so that data from different sources gets stored in a single repository, such as a database table within the Data Warehouse. The Data Warehouse also becomes the reference for the DSS to search for correlation data and the correlations will be stored in the Data Warehouse.

3.5.1 Component Introduction

Data retrieval, from a JNDMS perspective, is a three-step process consisting of data collection, data transformation and data storage. The Data Warehouse is the third step in this process, which consists of a data store of large volumes of multi-source data. The Data Warehouse component for the JNDMS will allow the system to store and manage all data from multiple sources, including:

- EIM Systems
- SIM Systems
- Operations, Vulnerability and Safeguard Data

The Data Warehouse component will accumulate, manage, retrieve and replicate data as required for the JNDMS. This flexibility will provide data consistency and rapid data retrieval performance. Flexibility will be added with the ability to add additional data sources through the Extract Transform Load (ETL) tool as part of the Data Transformation component.

3.5.2 Component Description

Definition of a data model is a key aspect of the Data Warehouse component. In conjunction with the development of the Data Transformation component, a data model will be defined that normalizes all the incoming data into a common representation. A key aspect of this, from the Data Warehouse perspective, is that the data model will define the performance and efficiency of the services deployed in the Data Warehouse. Therefore, data management and data retrieval objectives must be considered in the development of the data model.

The Data Warehouse component will be enhanced through the adoption of an entity relationship data modeling tool to analyze data from input sources and to forward engineer the warehouse. Once the Data Warehouse model is defined and instantiated, then the incoming data from operational applications will be transformed and moved into the Data Warehouse environment, thus allowing the data to be leveraged as information. The Data Warehouse will be augmented by collecting source data from interim data repositories from lower-level components.

These repositories will be made available with the use of the ETL tool in the Data Transformation component. This will allow for faster analytical processing and decision-making. The transformed data will be stored within the warehouse database and will be available for further analysis.

3.5.3 Component Technologies

The purpose of a database in the JNDMS is to be used as both a storage place for commands and messages, but also to allow for history and recoverability. In order to fulfill the need for good recoverability, we need the capability to save the commands and messages that are passed from the user and the JNDMS, and messages that are passed internal to the JNDMS between applications. Furthermore, in the case of a system failure, we must be able to recover the last set of commands and messages, and continue processing, with no loss of data.

The question arises of how this data store can be implemented, whether it should be embedded into a Control and Management application, or set up in such a way that other applications can access it. As most of the JNDMS will use this system to store messages, it should be set up in such a way that other applications can access the data store. In this case, a client server style of access is beneficial, as it allows all of the applications that require access to the data store to connect and have the connection maintained by the server.

Although a fully relational database system is ideal, we have the further need that this system must be efficient, as it cannot interfere with other applications that are running, or interfere with the processing capabilities of the computer in such a way that our timing requirements cannot be met. Trade-offs will be made between full normalization of the database to reduce data redundancy and less normalized forms to improve speed.

To implement a reliable storage solution, a relational database solution has been selected, which will use the Oracle Relational Database Management System (RDBMS) and a simple database schema. As recoverability is a high priority requirement, Oracle RDBMS is ideal. Most database management systems contain the capability to complete their own error correction and recovery, simplifying the disaster recovery for the JNDMS. The RDBMS further has the capabilities to handle high loading and balancing the query load, in order to efficiently provide a data storage solution that will not affect JNDMS operations.

Among the best-of-breed technologies that MDA has identified as candidates for this component in the JNDMS, the leading technology is Oracle Enterprise Server 10g. Oracle is an industry leader in database management systems. Their databases have built-in features for integrity checking, backups and replication. Oracle databases are currently deployed throughout the Canadian Forces (CF) and IT support personnel are abundant.

3.5.3.1 Oracle 10g Features Relevant to JNDMS

The following Oracle 10g features are relevant to JNDMS and described in the following sections:

- Performance – history and features
- Referential Integrity
- ODBC, JDBC, SQL*Net
- XML Handling
- Replication
- Distribution
- Grid Computing
- Backup and Recovery

3.5.3.2 Performance

Oracle has been an industry leading product for years and over time, has implemented several features to enhance the performance of applications running on an Oracle database. Oracle supports both B-Tree and Bitmap indexes to enhance query performance. The Oracle Optimizer allows for both rule based and cost based optimizations, which will allow the database to perform operations in using statistics of the actual database being worked against.

Oracle has several advanced tools to enable developers and administrators to get the best performance possible out of their database. The Explain Plan feature shows a developer how the Structured Query Language (SQL) will access the data and aid the developer in tuning the SQL to achieve optimum performance. Administrators have a full set of configuration parameters that can be used to tune database file organization and memory allocation. Further, Oracle has implemented automatic tuning capabilities that allow the database to tune parameters over time as the database matures.

3.5.3.3 Data Integrity

Oracle has implemented a full range of integrity features in the form of primary keys, unique keys, foreign keys and check constraints for over 10 years now. In addition, Oracle has implemented several features to allow more complex integrity to be enforced in the database. The database supports “pre”, “post” and “instead of” triggers allowing execution of more complex functionality and even an “Object-Relational” capability.

Further, the database supports both PL/SQL (a procedural extension to SQL) and JSQL (a Java based extension of SQL) within the database itself giving powerful coding capability as part of the database.

3.5.3.4 Connectivity

Oracle supports several connection types, including ODBC, JDBC and their own SQL*Net. JDBC supports the use of Java for database applications, providing the most flexible, portable and extendable technology for enterprise application development. For more information on the use of Java with Oracle technologies see <http://www.oracle.com/technology/tech/java/index.html>.

3.5.3.5 Data Transformation

Oracle provides an integrated tool for data transformation. Oracle XML Gateway uses a meta-data driven approach to dynamically generate outbound message and consume inbound messages. Stored in the meta-data repository is information about the trading partner, message maps, process controls, data transformation rules, syntax and semantic validation rules. You can change the way a message is processed simply by changing the meta-data in the repository. Changes can be made using user interfaces, which in turn will update the repository. The changes take effect at run-time without any code modifications.

3.5.3.6 Data Replication

Peer systems can be developed as a fully redundant capability, or each may take a portion of the work share. The detailed requirements for full and partial data replication with peer systems will be identified in phase 1, but many of these requirements, including the development and maintenance of peer systems profiles and policies, may be satisfied by the built-in capabilities and advanced replication features of the recommended Oracle Enterprise RDBMS. The development of these peer system profiles and policies will define the access controls and mechanisms required for the JNDMS to exchange with peer systems.

A hierarchy structure could also be supported where an instance of JNDMS would summarize its findings and report the summarized data to a higher level JNDMS. In this model, a lower level JNDMS becomes a "sensor" in the same fashion as the other sensors employed on the network and report to a JNDMS. This model implies a containment property in the relationship regardless of whether or not subordinate JNDMS instances exist at the same classification level as the parent system. This type of structure would require a different replication/distribution scheme from the one required for a peer-to-peer structure.

Finally, the two structures could be combined with peers dividing the workload at the lower levels and then each reporting to a higher node in the hierarchy.

3.5.3.7 Data Distribution

Oracle has great support for communicating between Oracle databases using distributed SQL, while maintaining location transparency and data integrity. To address the needs of applications that access non-Oracle system, Oracle offers Oracle Transparent Gateways and Generic Connectivity.

3.5.3.8 Grid Computing

The next extension to Data Distribution is Grid Computing. Oracle Database 10g includes a variety of Grid Computing features that provide efficient utilization of enterprise resources including server virtualization, storage virtualization, information virtualization and support for structured and non-structured data. In addition, Oracle provides a suite of integrated products that support grid computing, such as the Oracle Application Server and the Oracle Enterprise Manager.

3.5.3.9 Backup and Recovery

The JNDMS Data Warehouse will require an advanced replication capability. Replicating data as a backup not only protects JNDMS data contained within the warehouse, but also protects the investment of time and resources required to create and structure the data contained within that warehouse. Retaining historical versions of the data will also help the JNDMS to recover from errors, to facilitate traceability of information if required, and to protect the information if disaster recovery is required. The replication components of the Data Warehouse will enable the JNDMS to define backup policies by specifying critical data. The Oracle RDBMS has several powerful backup and recovery facilities including hot backups, incremental backups, full backups, file system based backups, database exports and more.

3.5.3.10 Database Access Control

The majority of the access controls will be enforced at the JNDMS Services level of the architecture. Users will not log into the database directly, instead users will log into the application and the application will manage the system access.

In addition, the Data Warehouse will incorporate a number of security features to ensure the confidentiality, integrity, and availability of the data.

Access control of data will be implemented using combinations of the following strategies:

1. Partitioning secure data from non-secure data, or sensitive data from non-sensitive data. This will allow access control at the systems level. When the secure data is classified, it will be as per the Security Orders for Classified Information Systems, Chapter 4 (Operational Security Standards) [R-11].
2. Implementing user-level access control within the database management system.
3. Implementing user-level access control within the middleware for the portal application.
4. Use of “peer systems profiles” to dictate what data views are synchronized to which JNDMS peers; for example one profile for a CF to CF synchronization, and another for a CF to North Atlantic Treaty Organization (NATO) synchronization.

Encryption will be used as appropriate using standards based mechanisms:

1. Encapsulation of JDBC/ ODBC calls over the network using SSL or Secure Internet Profile (IPSEC) tunnelling.
2. Appropriate use of on-disk encryption using encrypting file systems where host Threat Risk Assessment dictates.

Integrity checking mechanisms are built into the underlying relational database management technology and additional mechanisms, such as inclusion of calculated message digests on input files, can be included in the data schema as well.

Backup and recovery mechanisms that exist within the database technology will be leveraged both technically and through CONOPS documentation, including how and when to use full or partial restores.

Typical database access control is handled by several means. The first implementation is via the database user. Every connection to the database is done via a database username and password. Table 15 shows the different classes of database users that will be created.

Table 15: JNDMS Database Users

User	Description
Admin Users (sys, system)	There are administrative users created by default for the Oracle database. These will remain in place such that database administrators can perform the necessary administrative and maintenance functions using standard COTS tools.
JNDMS Owner (jndms)	The JNDMS Owner user is the owner of the schema objects and the data. This account will be used by application developers/ maintainers to create and alter the schema objects, changing the definition of the object stored in the database. By default, this user has full Create/Read/Update/Delete (CRUD) access to the actual data.
JNDMS User(s)	A JNDMS user database account references the JNDMS owner's objects. This type of user will have the ability to update the data but no ability to alter the schema definition. There can be several different users created with different levels of access if this is deemed necessary. Typically, these users will only access the database via the specific application, in this case JNDMS. Often, this username and password is contained in a configuration file and is not directly related to an operator's user account.

The second form of database access control is provided by the application. The rules coded within the application restrict the access and what data is available and at the same time are able to navigate through the database structure to produce/present the desired results.

3.5.4 Data Model

3.5.4.1 Data Modelling Process

An entity-relationship data modeling solution (as shown in Figure 3-3) will support the building of comprehensive, robust data structures within the Data Warehouse. The design layer architecture allows an organization to create data architectures that support the organization's processes by aligning the models, from the conceptual level to the physical implementation. The JNDMS Data Warehouse model can be created using an entity-relationship data modeling solution at the conceptual level, i.e., identify the conceptual model artefacts and high-level subject areas. This conceptual model can then be used as a basis to design the logical data structures. As required, logical sub-models are derived from the overarching model that include the appropriate logical model objects, but incorporate more detail as required from the various modules.

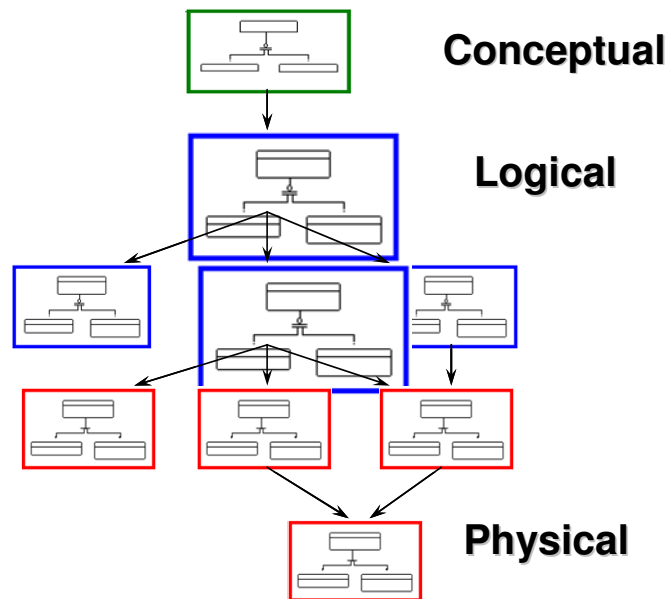


Figure 3-3: Designing the Data Warehouse from an Entity-Relationship Data Model

Finally, the physical data models and associated database structures are created and deployed. Design layer architecture supports model synchronization as models and requirements change with time.

3.5.4.2 JNDMS

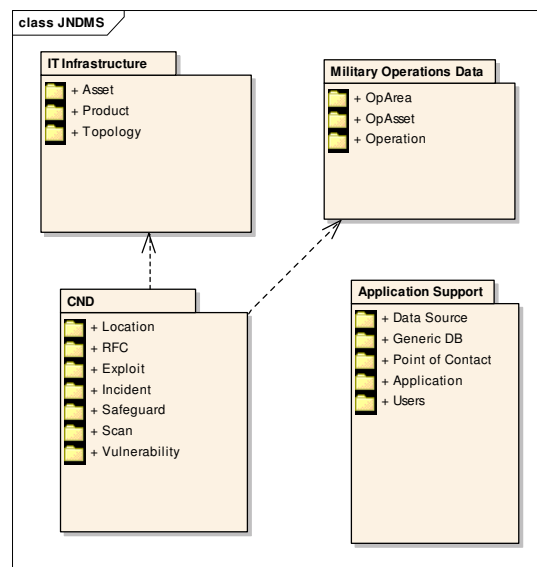
The Data Model is based upon the requirements that will consider the data required to present to the user, the data available from the sensors and the data to be processed by the DSS. The JNDMS data model will be unique in that it will correlate data from the distinct domains, namely IT infrastructure, information/data security, and military operations.

The area in which the JNDMS will truly make progress will be in relating all these models together. By relating the data from all three domains, along with temporal attributes, Commanders will be able to identify the real world impact of the various threats. Armed with this knowledge, CND resources can be applied and strategies developed, based upon priorities with an understanding of the impact of a potential action or inaction.

Data that will be stored in the Data Warehouse includes the following:

1. IT Infrastructure and Services Data
2. Military Operations Data
3. Security Events Data
4. DSS Rules
5. Incident Data, DSS Assessments and Defensive Posture Conclusions
6. Profiles for Policies for Data Sharing (import and export) with Peers
7. Imported Datasets from External Peers
8. Maintain Accurate References to the Sources of Data
9. Linkage Information to External Data Sources, which may hold accessible information that is relevant to the JNDMS
10. Interrelationships
11. Metadata for all of the above

JNDMS - (Logical diagram)

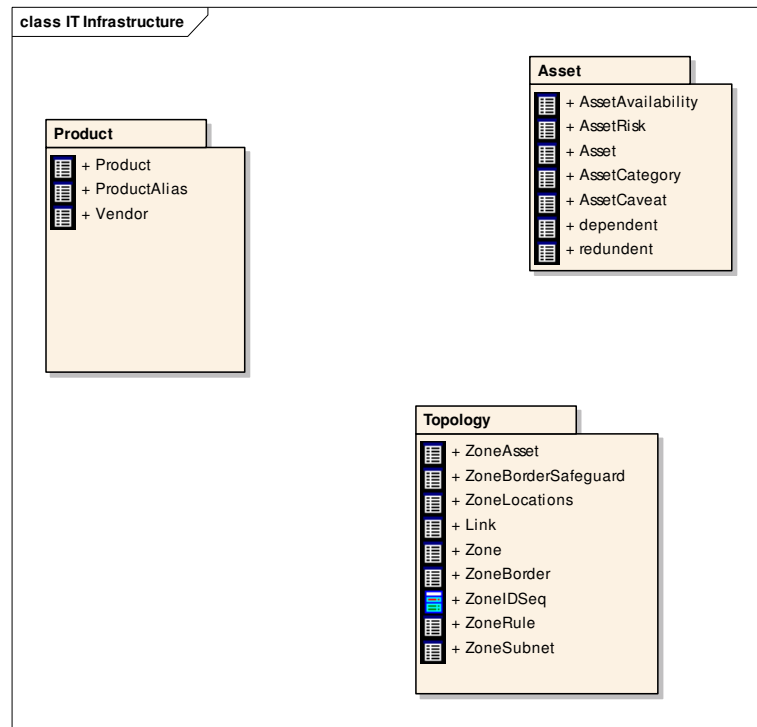


3.5.4.3 IT Infrastructure

The IT Infrastructure data represents the network. This data deals first of all with the hardware within the network but as a critical extension: the services that the hardware offers. The IT Infrastructure data model also models the dependencies that exist with the Infrastructure. This ensures that any given service instance can go through a multi-link analysis to identify all assets (and by extension all operations) affected by a given vulnerability.

The IT Infrastructure is divided into 3 sub-packages, Topology, Asset and Product.

IT Infrastructure - (Logical diagram)



Asset

The Asset package contains entities related to the assets. Assets may be hardware, software, services etc.

Asset - (Logical diagram)

Asset

Type: **Class**

Stereotype: «table»

Notes:

A DND CM Database (driven by RFCs) should be the source of this data.

Incidents will be raised against the infrastructure via the network discovery sensors without having to do any comparison to or impact on the asset table. Technically, the network discovery tool's database should be set to match the DND database at time T0. That way, all discovered items (outside of this list) will be flagged.

The asset table uses an artificial unique identifier as there are several different attributes that may or may not be used to support uniquely identifying the asset.

A "Configuration" table has been considered that would hold general configuration rules for assets. It has not been included at this time as there has not been a method identified that would verify the configuration is enforced.

Attributes

Name	Type	Length	Precision	Scale	Notes
serviceID	NUMBER	0	12	0	Unique identifier.
created	TIMESTAMP	0	0	0	Timestamp of when the record was created.
modified	TIMESTAMP	0	0	0	Timestamp of last modification.
enabled	CHAR	1	0	0	Y or N. Indicates whether or not the asset is online or currently disabled (i.e., cold spare in storage).
status	VARCHAR2	8			The status value represents the status of having the asset added to the approved baseline. Values include: New Pending Approved Denied
agent	CHAR	1			The agent flag is set to 'Y' if this asset is considered an agent to JNDMS, meaning it provides reports to JNDMS.

DN0678
Issue 4/2: 06 February 2012

Name	Type	Length	Precision	Scale	Notes
importance	NUMBER		3	0	Summary of asset's importance values (inherited from the opAsset importance values).
type	VARCHAR2	20	0	0	Asset type. This could also be traced through the product information however, the data here, in the asset trumps the link via product. Allowable values include: hardware user service system service software service capability
category	VARCHAR2	80	0	0	Asset category (i.e., Email, Workstation). This could also be traced through the product information however, the data here, in the asset trumps the link via product.
assetCount	NUMBER		8	0	The number of assets this asset entry represents. For example, 500 installs of Windows XP
layer	CHAR	1	0	0	Allowable Values are 1-7 representation of the OSI layer.
confidentialityValue	VARCHAR2	14	0	0	Confidentiality value of this asset. Values include critical, high, medium, low or not applicable.
integrityValue	VARCHAR2	14	0	0	Integrity value of this asset. Values include critical, high, medium, low or not applicable.
availabilityValue	VARCHAR2	14	0	0	Confidentiality value of this asset. Values include critical, high, medium, low or not applicable.
confidentialityStatus	VARCHAR2	15			The current status of the current confidentiality.
integrityStatus	VARCHAR	15			The current value of the integrity of the asset.

DN0678
Issue 4/2: 06 February 2012

Name	Type	Length	Precision	Scale	Notes
availabilityStatus	VARCHAR2	15			The current value of the availability of the asset.
heartRate	NUMBER	0	6	0	Count of seconds between expected heart beats. Null indicates no pulse expected. Heartbeats are stored as events in the incident table.
name	VARCHAR2	255	0	0	Asset name from whatever source is available such as host name, DNS name, etc.
assetTag	VARCHAR2	255	0	0	Asset tag id from local asset management system. Usually located on a sticker on the machine.
vendor	VARCHAR2	255	0	0	Asset vendor.
product	VARCHAR2	255	0	0	Asset Model Number.
version	VARCHAR2	255	0	0	Asset version.
serialNumber	VARCHAR2	255	0	0	Asset serial number.
description	VARCHAR2	2000	0	0	Description of the asset.
ifIndex	NUMBER		3	0	the index for the if
IP	VARCHAR2	64	0	0	IP of the asset.
MACAddress	VARCHAR2	64	0	0	MAC Address.
securityLevel	NUMBER	0	1	0	Security Classification Level.
capacity	NUMBER	0	12	5	The throughput limit of the asset. For example, the speed of a network link.
capacityUnit	VARCHAR2	20	0	0	The unit of measure for the capacity attribute. For example, megabits per second.
networkHost	NUMBER	0	12	0	This is a key back to the asset table. If this key is filled, it implies that this asset is a network interface for the asset being referenced. This relationship is a convenience relationship and to some extent is a duplication of the dependent relationship that would exist between these assets.

DN0678
Issue 4/2: 06 February 2012

Name	Type	Length	Precision	Scale	Notes
source	VARCHAR2	255	0	0	This identifies the source of the asset information, which may even be a specific instance of a network discovery tool.
sourceUID	VARCHAR2	255	0	0	A character representation of the unique identifier as provided by the source.
sourceName	VARCHAR2	255	0	0	The name associated with the asset as identified by the source.
sourceLabel	VARCHAR2	255	0	0	The label used for a user interface on the data source.
decIP	NUMBER	0	38	0	The decimal IP is the integer representation of the IP (IPv4).
notes	VARCHAR2	2000			Notes about the asset. May be entered by the system or by the operator.
pathStatus	VARCHAR2	30			Used in calculating the path for the DSS memory model
dependencyStatus	VARCHAR2	30			Used in calculating the DSS in memory model
dsscvalue	NUMBER		8	2	DSS raw C value (corresponds to text in .CONFIDENTIALITYVALUE)
dssivalue	NUMBER		8	2	DSS raw I value (corresponds to text in .INTEGRITYVALUE).
dssavalue	NUMBER		8	2	DSS raw A value (corresponds to text in .AVAILABILITYVALUE)
DSSEXPLANATION	CLOB				An explanation for the DSS's calculations

Connections

Connector	Source	Target
<u>Association</u> (serviceID = serviceID) Source -> Destination	Asset:AssetAvailability	Asset:Asset
<u>Association</u> (serviceID = serviceID) Source -> Destination	Asset:AssetRisk	Asset:Asset
<u>Association</u>	Topology:ZoneSubnet	Asset:Asset

DN0678
Issue 4/2: 06 February 2012

Connector	Source	Target
Source -> Destination		
<u>Association</u> (serviceID = serviceID) Source -> Destination	Safeguard:SafeguardProtection	Asset:Asset
<u>Association</u> (ServiceID = serviceID) Source -> Destination	Point of Contact:AssetPOC	Asset:Asset
<u>Association</u> (serviceID = serviceID) Source -> Destination	Topology:ZoneAsset	Asset:Asset
<u>Association</u> (affectedAsset = serviceID) Source -> Destination	Incident:Incident	Asset:Asset
<u>Association</u> (serviceID = serviceID) Source -> Destination	Safeguard:SafeguardImplementation	Asset:Asset
<u>Association</u> (assetB = serviceID) Source -> Destination	Topology:Link	Asset:Asset
<u>Association</u> (targetServiceID = serviceID) Source -> Destination	Scan:ScanTarget	Asset:Asset
<u>Association</u> (serviceID = serviceID) Source -> Destination	OpAsset:OpAsset	Asset:Asset
<u>Association</u> (serviceID = serviceID) Source -> Destination	Scan:Scan	Asset:Asset
<u>Association</u> (serviceID = serviceID) Source -> Destination	Vulnerability:VulnerabilityInstance	Asset:Asset
<u>Dependency</u> Source -> Destination	Presentation Visualization Alerting:Query - IT Infrastructure affected by new Vulnerability	Asset:Asset
<u>Association</u> (networkHost = serviceID) Source -> Destination	Asset:Asset	Asset:Asset

Connector	Source	Target
<u>Association</u> (sensorID = serviceID) Source -> Destination	Incident:Incident	Asset:Asset
<u>Association</u> (assetA = serviceID) Source -> Destination	Topology:Link	Asset:Asset
<u>Association</u> (serviceID = serviceID) Source -> Destination	Asset:AssetCaveat	Asset:Asset
<u>Association</u> (discoveredServiceID = serviceID) Source -> Destination	Scan:ScanHit	Asset:Asset
<u>Association</u> (serviceID = serviceID) Source -> Destination	Topology:ZoneBorder	Asset:Asset
<u>Association</u> (serviceID = serviceID) Source -> Destination	Asset:redundant	Asset:Asset
<u>Association</u> (redundant = serviceID) Source -> Destination	Asset:redundant	Asset:Asset
<u>Association</u> (type = type category = category) Source -> Destination	Asset:Asset	Asset:AssetCategory
<u>Association</u> (ITID = serviceID) Source -> Destination	Asset:dependent	Asset:Asset
<u>Association</u> (connectionID = serviceID) Source -> Destination	Asset:dependent	Asset:Asset
<u>Association</u> (vendor = vendor product = product version = version) Source -> Destination	Asset:Asset	Product:Product

Operations

Method	Parameter
--------	-----------

Method	Parameter
FK_Asset_Asset()	<u>NUMBER</u> networkHost [in]
FK_Asset_AssetCategory()	<u>VARCHAR2</u> type [in] <u>VARCHAR2</u> category [in]
FK_Asset_Product()	<u>VARCHAR2</u> vendor [in] <u>VARCHAR2</u> product [in] <u>VARCHAR2</u> version [in]
PK_Asset()	<u>NUMBER</u> serviceID [in]
TRG_PreInsert_Asset before insert()	
CHK_ASSETCAT_STATUS()	
TRG_Asset_ChangeLog after insert or update()	
CHK_AGENT()	
TRG_PreDelete_Asset before delete()	
TRG_IPUpdate_Asset before insert or update ()	

AssetAvailability

Type: **Class**

Stereotype: «table»

Notes:

This will track the availability of the asset over time.

Attributes

Name	Type	Length	Precision	Scale	Notes
serviceID	NUMBER		12	0	The serviceID of the asset for which this availability applies
StartDate	TIMESTAMP				The date at which the asset is available
StopDate	TIMESTAMP				The date at which the asset is no longer available
IsExclusive	VARCHAR2	1			Y/N whether or not it is exclusively up - i.e., available to only one operation at a time.

Connections

Connector	Source	Target
<u>Association</u> (serviceID = serviceID) Source -> Destination	Asset:AssetAvailability	Asset:Asset

Operations

Method	Parameter
FK_AssetAvailability_Asset()	<u>NUMBER</u> serviceID [in]

AssetCategory

Type: **Class**

Stereotype: «table»

Notes:

The Asset Category provides an attribute to ease operator searching and filtering operations.

Attributes

Name	Type	Length	Precision	Scale	Notes
type	VARCHAR2	20			The type of asset (i.e., primary hardware, secondary hardware, system service, user service, network service, capability).
category	VARCHAR2	80	0	0	Asset category (i.e., Email, Workstation)
created	TIMESTAMP	0	0	0	Timestamp when record was created.
modified	TIMESTAMP	0	0	0	Timestamp of last modification.

Connections

Connector	Source	Target
Association (type = type category = category) Source -> Destination	Product:Product	Asset:AssetCategory
Association (type = type category = category) Source -> Destination	Asset:Asset	Asset:AssetCategory

Operations

Method	Parameter
PK_AssetCategory()	VARCHAR2 type [in] VARCHAR2 category [in]
TRG_PreInsert_AssetCategory before insert()	

3.5.4.3.1.4 AssetCaveat

Type: **Class**

Stereotype: «table»

Notes:

The list of countries (caveats) for the asset.

Attributes

Name	Type	Length	Precision	Scale	Notes
serviceID	NUMBER	0	12	0	Asset Service ID.
caveat	VARCHAR2	3	0	0	Country code (caveat).
created	TIMESTAMP	0	0	0	Timestamp when record was created.
modified	TIMESTAMP	0	0	0	Timestamp of last modification.

Connections

Connector	Source	Target
<u>Association</u> (serviceID = serviceID) Source -> Destination	Asset:AssetCaveat	Asset:Asset

Operations

Method	Parameter
FK_AssetCaveats_Asset()	<u>NUMBER</u> serviceID [in]
PK_AssetCaveats()	<u>NUMBER</u> serviceID [in] <u>VARCHAR2</u> caveat [in]
TRG_PreInsert_AssetCaveat before insert()	

AssetRisk

Type: **Class**

Stereotype: «table»

Notes:

This will store data calculated by the DSS

Attributes

Name	Type	Length	Precision	Scale	Notes
serviceID	NUMBER		8	2	Foreign key
RISKC	NUMBER		8	2	default risk score normalized to a 0-100 range
RISKI	NUMBER		8	2	default risk score normalized to a 0-100 range
RISKA	NUMBER		8	2	default risk score normalized to a 0-100 range
DSSRISKC	NUMBER		8	2	default DSS raw risk score
DSSRISKI	NUMBER		8	2	default DSS raw risk score
DSSRISKA	NUMBER		8	2	default DSS raw risk score
DSSRISK	NUMBER		8	2	default DSS raw “unified” risk score
P_ATTACK	NUMBER		8	2	default DSS estimated “probability of attack attempt”
P_EXPLOITABILITY	NUMBER		8	2	default DSS estimated “probability of exploitation if attacked”
RISKC_1D	NUMBER		8	2	1 day risk score normalized to a 0-100 range
RISKI_1D	NUMBER		8	2	1 day risk score normalized to a 0-100 range
RISKA_1D	NUMBER		8	2	1 day risk score normalized to a 0-100 range

DN0678
Issue 4/2: 06 February 2012

Name	Type	Length	Precision	Scale	Notes
DSSRISKC_1D	NUMBER		8	2	1 day DSS raw risk score
DSSRISKI_1D	NUMBER		8	2	1 day DSS raw risk score
DSSRISKA_1D	NUMBER		8	2	1 day DSS raw risk score
DSSRISK_1D	NUMBER		8	2	1 day DSS raw “unified” risk score
P_ATTACK_1D	NUMBER		8	2	1 day DSS estimated “probability of attack attempt”
P_EXPLOITABILITY_1D	NUMBER		8	2	1 day DSS estimated “probability of exploitation if attacked”
RISKC_1W	NUMBER		8	2	1 week risk score normalized to a 0-100 range
RISKI_1W	NUMBER		8	2	1 week risk score normalized to a 0-100 range
RISKA_1W	NUMBER		8	2	1 week risk score normalized to a 0-100 range
DSSRISKC_1W	NUMBER		8	2	1 week DSS raw risk score
DSSRISKI_1W	NUMBER		8	2	1 week DSS raw risk score
DSSRISKA_1W	NUMBER		8	2	1 week DSS raw risk score
DSSRISK_1W	NUMBER		8	2	1 week DSS raw “unified” risk score
P_ATTACK_1W	NUMBER		8	2	1 week DSS estimated “probability of attack attempt”
P_EXPLOITABILITY_1W	NUMBER		8	2	1 week DSS estimated “probability of exploitation if attacked”
RISKC_1M	NUMBER		8	2	1 month risk score normalized to a 0-100 range
RISKI_1M	NUMBER		8	2	1 month risk score normalized to a

DN0678
Issue 4/2: 06 February 2012

Name	Type	Length	Precision	Scale	Notes
					0-100 range
RISKA_1M	NUMBER		8	2	1 month risk score normalized to a 0-100 range
DSSRISKC_1M	NUMBER		8	2	1 month DSS raw risk score
DSSRISKI_1M	NUMBER		8	2	1 month DSS raw risk score
DSSRISKA_1M	NUMBER		8	2	1 month DSS raw risk score
DSSRISK_1M	NUMBER		8	2	1 month DSS raw “unified” risk score
P_ATTACK_1M	NUMBER		8	2	1 month DSS estimated “probability of attack attempt”
P_EXPLOITABILITY_1M	NUMBER		8	2	1 month DSS estimated “probability of exploitation if attacked”

Connections

Connector	Source	Target
<u>Association</u> (serviceID = serviceID) Source -> Destination	Asset:AssetRisk	Asset:Asset

Operations

Method	Parameter
FK_AssetRisk_Asset()	<u>NUMBER</u> serviceID [in]
PK_AssetRisk()	<u>NUMBER</u> serviceID [in]

3.5.4.3.1.6 dependent

Type: **Class**

Stereotype: «table»

Notes:

The dependent table shows the asset dependency. This web of data will support link analysis to determine which operations are affected by a given asset.

Attributes

Name	Type	Length	Precision	Scale	Notes
ITID	NUMBER	0	12	0	The Asset ID.
connectionID	NUMBER	0	12	0	Asset ID that the ITID is dependent upon.
created	TIMESTAMP	0	0	0	Timestamp record was created.
modified	TIMESTAMP	0	0	0	Timestamp last modified.

Connections

Connector	Source	Target
<u>Association</u> (ITID = serviceID) Source -> Destination	Asset:dependent	Asset:Asset
<u>Association</u> (connectionID = serviceID) Source -> Destination	Asset:dependent	Asset:Asset

Operations

Method	Parameter
FK_connectionID_Asset()	<u>NUMBER</u> connectionID [in]
FK_LogicalConnections_Asset()	<u>NUMBER</u> ITID [in]
PK_LogicalConnections()	<u>NUMBER</u> ITID [in] <u>NUMBER</u> connectionID [in]
TRG_PreInsert_dependent before insert()	

3.5.4.3.1.7 **redundant**

Type: **Class**

Stereotype: **«table»**

Notes:

The redundant table models redundant assets on the system. Assuming that dependency and redundancy of assets can be broken into "and" and "or" relationships, redundant provides the "or" relationship and "dependent" defines the "and" relationship.

For example, assume the following asset "a" is dependent upon (("b" or "c") and "d"). To model this we will have to create new asset "bc". Table redundant will have records ("bc","b") and ("bc","c"). Table dependent will have records ("a", "bc") and ("a","d").

To further enhance this relationship, the "loadShare" attribute indicates whether or not the capacities of these redundant assets can be aggregated. The capacity attributes are part of the asset table.

Attributes

Name	Type	Length	Precision	Scale	Notes
serviceID	NUMBER		12	0	Foreign key to the Asset Table
redundant	NUMBER		12	0	Foreign key to the Asset table. This is the Asset that is redundant
loadShare	CHAR	1	0	0	Do the redundant assets provide a "load sharing" capability. Valid values are "Y" or "N".
created	TIMESTAMP	0	0	0	Timestamp record was created.
modified	TIMESTAMP	0	0	0	Timestamp of last modification.

Connections

Connector	Source	Target
<u>Association</u> (serviceID = serviceID) Source -> Destination	Asset:redundant	Asset:Asset
<u>Association</u> (redundant = serviceID) Source -> Destination	Asset:redundant	Asset:Asset

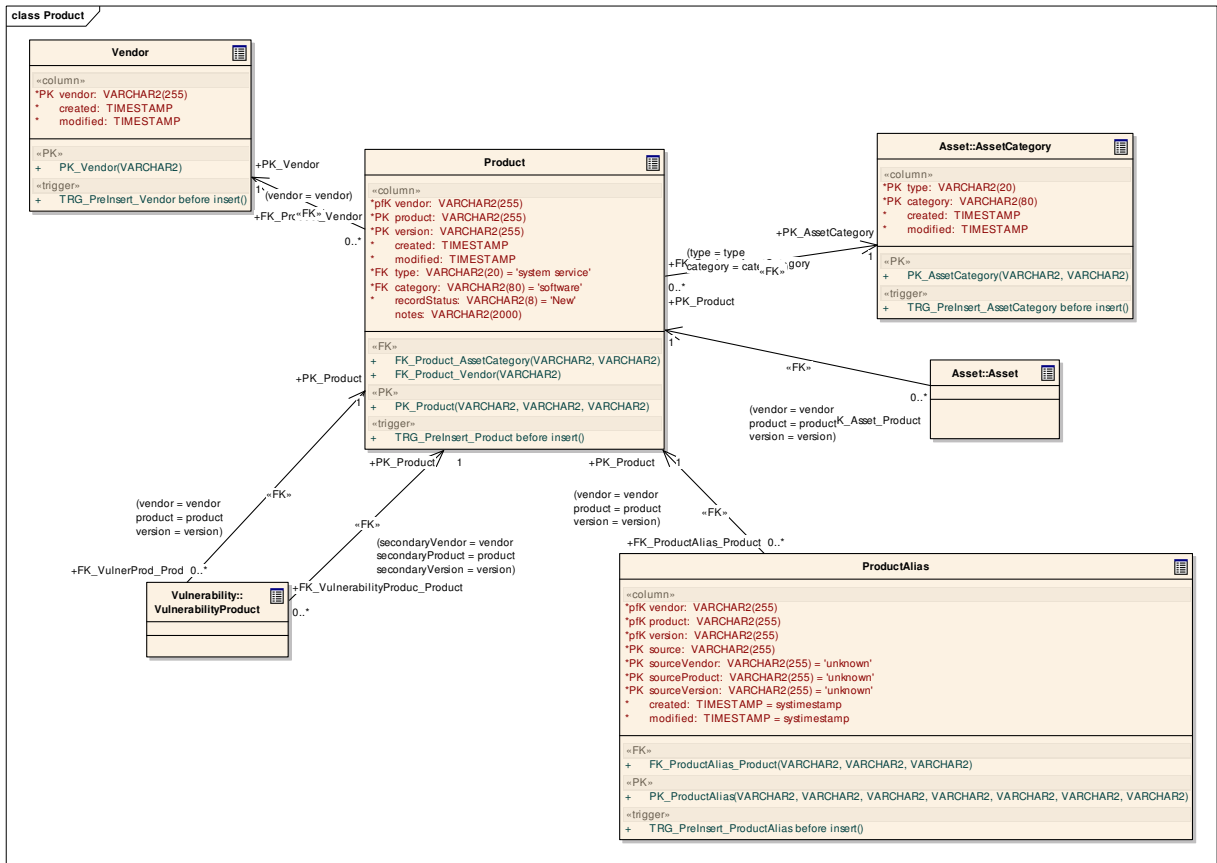
Operations

Method	Parameter
FK_parent()	<u>NUMBER</u> redundant [in]
FK_redundant()	<u>NUMBER</u> serviceID [in]
PK_redundant()	<u>NUMBER</u> serviceID [in] <u>NUMBER</u> redundant [in]
CHK_loadShare()	
TRG_PreInsert_redundent before insert()	

3.5.4.3.2 Product

The Product package models the normalized list of products and vendors. Data may be managed via the JNDMS UI or by downloading the Product Dictionary from the NVD website. The list may include entries not available from the NVD.

Product - (Logical diagram)



3.5.4.3.2.1 Product

Type: **Class**

Stereotype: «table»

Notes:

The Product table is a look-up of normalized products. It is a child of the vendor list. The data may be filled from a download of the NVD product dictionary or via the UI.

Attributes

Name	Type	Length	Precision	Scale	Notes
vendor	VARCHAR2	255	0	0	Vendor name from the vendor table.
product	VARCHAR2	255	0	0	Product name.
version	VARCHAR2	255	0	0	Version identifier for the product.
created	TIMESTAMP	0	0	0	Timestamp of record creation.
modified	TIMESTAMP	0	0	0	Timestamp of last modification.
type	VARCHAR2	20			The type of asset this product is.
category	VARCHAR2	80	0	0	The asset category of this product.
recordStatus	VARCHAR2	8			Status of the product. Allowable values and meanings as follows: New - this product has been added due to a source not finding a match. To see details on the source, query the product alias records. Approved - This product data has been accepted into the normalized list. Pending - future Denied - future
notes	VARCHAR2	2000			Notes (either from processing or from the operator).

Connections

Connector	Source	Target
<p><u>Association</u>(secondaryVendor = vendor secondaryProduct = product secondaryVersion = version) Source -> Destination</p>	Vulnerability:VulnerabilityProduct	Product:Product
<p><u>Association</u>(vendor = vendor) Source -> Destination</p>	Product:Product	Product:Vendor
<p><u>Association</u>(vendor = vendor product = product version = version) Source -> Destination</p>	Vulnerability:VulnerabilityProduct	Product:Product
<p><u>Association</u>(vendor = vendor product = product version = version) Source -> Destination</p>	Product:ProductAlias	Product:Product
<p><u>Association</u>(type = type category = category) Source -> Destination</p>	Product:Product	Asset:AssetCategory
<p><u>Association</u>(vendor = vendor product = product version = version) Source -> Destination</p>	Asset:Asset	Product:Product

Operations

Method	Parameter
FK_Product_AssetCategory()	<u>VARCHAR2</u> type [in] <u>VARCHAR2</u> category [in]
FK_Product_Vendor()	<u>VARCHAR2</u> vendor [in]
PK_Product()	<u>VARCHAR2</u> vendor [in] <u>VARCHAR2</u> product [in] <u>VARCHAR2</u> version [in]
TRG_PreInsert_Product before insert()	

3.5.4.3.2.2 ProductAlias

Type: **Class**

Stereotype: «table»

Notes:

The product alias table provides a mapping between the JNDMS "normalized" product list and an alternative source's product data.

Attributes

Name	Type	Length	Precision	Scale	Notes
vendor	VARCHAR2	255	0	0	Vendor name
product	VARCHAR2	255	0	0	Product name.
version	VARCHAR2	255	0	0	Version
source	VARCHAR2	255	0	0	Source of the alternative product information
sourceVendor	VARCHAR2	255	0	0	The source vendor information. "unknown" is a magic value that is used to represent an absence of data from the source.
sourceProduct	VARCHAR2	255	0	0	The source product information. "unknown" is a magic value that is used to represent an absence of data from the source.
sourceVersion	VARCHAR2	255	0	0	The source version information. "unknown" is a magic value that is used to represent an absence of data from the source.
created	TIMESTAMP	0	0	0	Timestamp of record creation.
modified	TIMESTAMP	0	0	0	Timestamp of last modification.

Connections

Connector	Source	Target
<u>Association</u> (vendor = vendor product = product version = version) Source -> Destination	Product:ProductAlias	Product:Product

Operations

Method	Parameter
FK_ProductAlias_Product()	<u>VARCHAR2</u> vendor [in] <u>VARCHAR2</u> product [in] <u>VARCHAR2</u> version [in]
PK_ProductAlias()	<u>VARCHAR2</u> vendor [in] <u>VARCHAR2</u> product [in] <u>VARCHAR2</u> version [in] <u>VARCHAR2</u> source [in] <u>VARCHAR2</u> sourceVendor [in] <u>VARCHAR2</u> sourceProduct [in] <u>VARCHAR2</u> sourceVersion [in]
TRG_PreInsert_ProductAlias before insert()	

3.5.4.3.2.3 Vendor

Type: **Class**

Stereotype: «table»

Notes:

The Vendor table is a lookup of all product vendors. It is based upon the NVD Product Dictionary but may not be restricted to only that list. The data may be loaded via download from the NVD website OR via the user interface.

Attributes

Name	Type	Length	Precision	Scale	Notes
vendor	VARCHAR2	255	0	0	The vendor name.
created	TIMESTAMP	0	0	0	Timestamp of record creation.
modified	TIMESTAMP	0	0	0	Timestamp of last modification.

Connections

Connector	Source	Target
<u>Association</u> (vendor = vendor) Source -> Destination	Product:Product	Product:Vendor

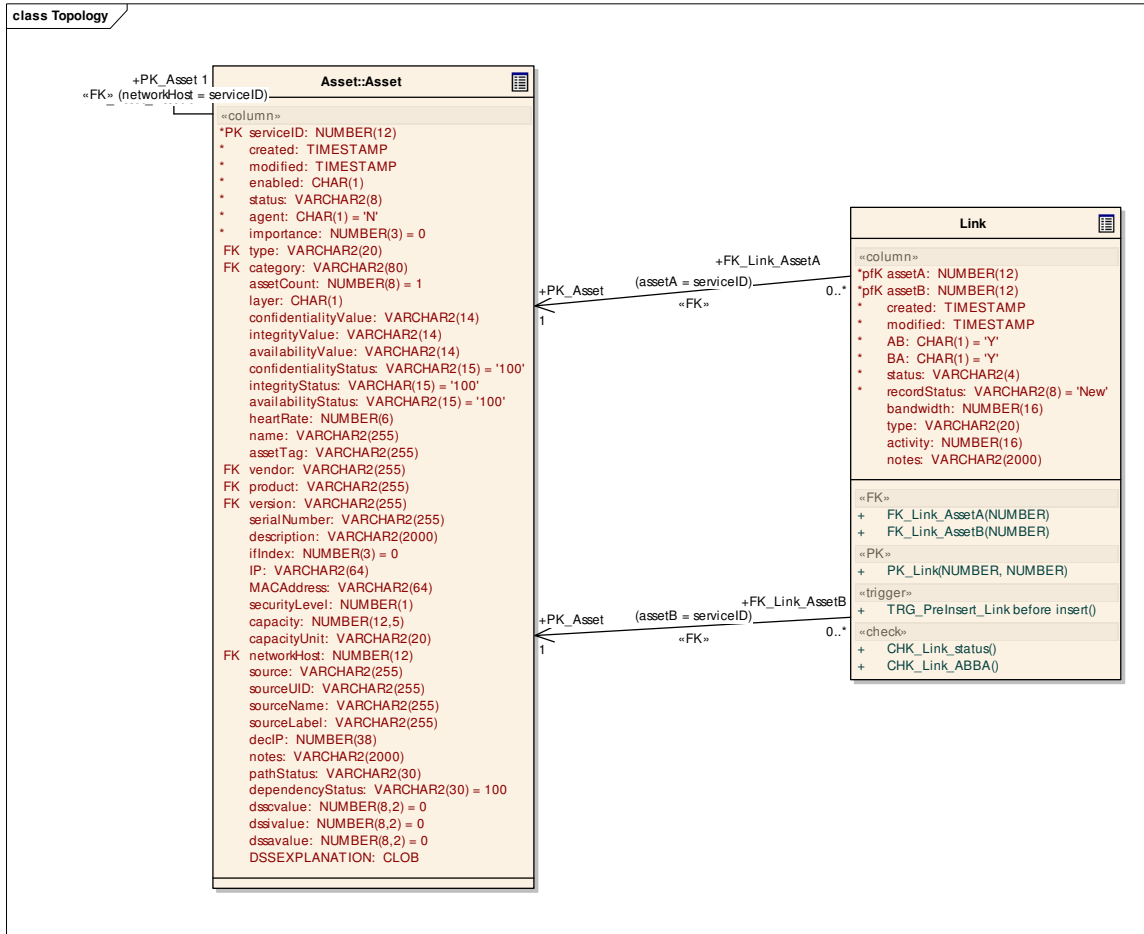
Operations

Method	Parameter
PK_Vendor()	<u>VARCHAR2</u> vendor [in]
TRG_PreInsert_Vendor before insert()	

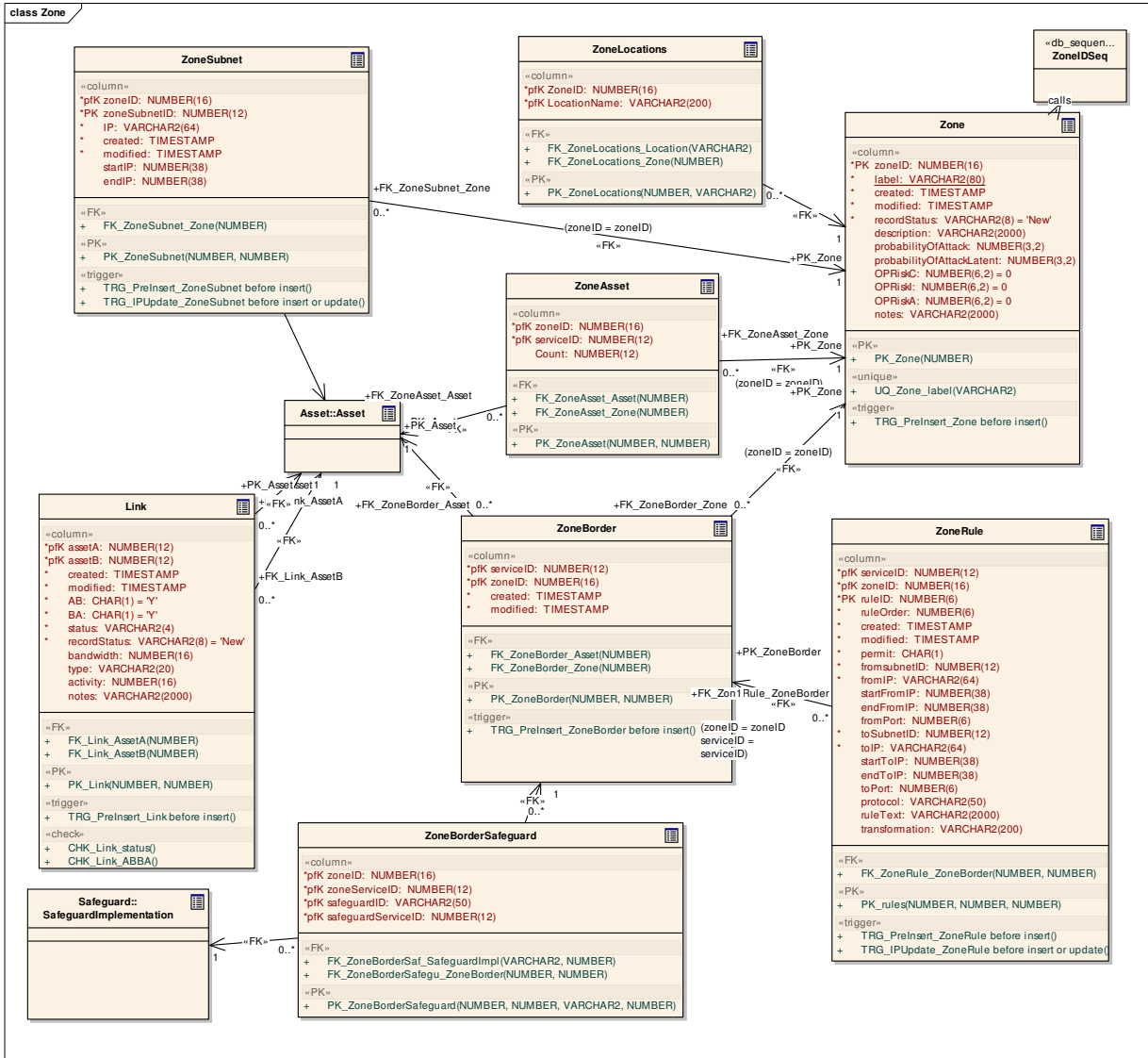
3.5.4.3.3 Topology

The Topology package contains entities related to the network topology.

Topology - (Logical diagram)



Zone - (Logical diagram)



3.5.4.3.3.1 Link

Type: **Class**

Stereotype: «table»

Notes:

The Link table holds all network links between assets. Currently, the only assets that can have a link must be a network interface and have an IP. These attributes exist on the asset table itself and therefore are not enforced at this time.

Attributes

Name	Type	Length	Precision	Scale	Notes
assetA	NUMBER	0	12	0	One side of the link.
assetB	NUMBER	0	12	0	Other side of the link.
created	TIMESTAMP	0	0	0	Timestamp of record creation.
modified	TIMESTAMP	0	0	0	Timestamp of last modification.
AB	CHAR	1	0	0	Y or N on whether or not link travels from A to B.
BA	CHAR	1	0	0	Y or N on whether or not the link travels from B to A.
status	VARCHAR2	4	0	0	up or down
recordStatus	VARCHAR2	8			Record status indicates the internal JNDMS handling of the data. It will queue operators to verify/rectify the data and it will indicate whether or not data is to be used by automated processing. Allowable values include: New- a new record that requires the attention of the operator. Pending - a record currently being research/reviewed. Approved - a record that is approved for use. approval may be done by automated processing or by the operator. Denied - A record that is to be retained by the system but is not be used.
bandwidth	NUMBER	0	16	0	Bits per second bandwidth.

DN0678
Issue 4/2: 06 February 2012

Name	Type	Length	Precision	Scale	Notes
type	VARCHAR2	20	0	0	Type of link: sat, radio, leased, point-to-point
activity	NUMBER	0	16	0	Bandwidth usage snapshot reading
notes	VARCHAR2	2000			Edit field for a memo of current issues or upcoming issues. (i.e., "the lease expires on Apr 15, 2008")

Connections

Connector	Source	Target
<u>Association</u> (assetB = serviceID) Source -> Destination	Topology:Link	Asset:Asset
<u>Association</u> (assetA = serviceID) Source -> Destination	Topology:Link	Asset:Asset

Operations

Method	Parameter
FK_Link_AssetA()	<u>NUMBER</u> assetA [in]
FK_Link_AssetB()	<u>NUMBER</u> assetB [in]
PK_Link()	<u>NUMBER</u> assetA [in] <u>NUMBER</u> assetB [in]
TRG_PreInsert_Link before insert()	
CHK_Link_status()	
CHK_Link_ABBA()	

3.5.4.3.3.2 Zone

Type: **Class**

Stereotype: «table»

Notes:

The zone is a combination of the assets providing the access control (such as a firewall), and a list of subnets. The zones are essentially created by a set of 1 or more firewall rules into a given set of 1 or more subnets.

Attributes

Name	Type	Length	Precision	Scale	Notes
zoneID	NUMBER	0	16	0	An artificial PK since the system will create the zones automatically.
label	VARCHAR2	80	0	0	Initially the system will provide some sort of label indicating the name of the firewall creating the zone. The user will be able to re label it to a more human readable name. The label is being forced as unique so any auto-inserts will have to handle the exception properly.
created	TIMESTAMP	0	0	0	Timestamp of record creation.
modified	TIMESTAMP	0	0	0	Timestamp of last record modification.
recordStatus	VARCHAR2	8			Record status indicates the internal JNDMS handling of the data. It will queue operators to verify/rectify the data and it will indicate whether or not data is to be used by automated processing. Allowable values include: New- a new record that requires the attention of the operator. Pending - a record currently being research/reviewed. Approved - a record that is approved for use. approval may be done by automated processing or by the operator. Denied - A record that is to be retained by the system but is not be used.
description	VARCHAR2	2000	0	0	A long textual description of the zone

DN0678
Issue 4/2: 06 February 2012

Name	Type	Length	Precision	Scale	Notes
					that may be provided by the user.
probabilityOfAttack	NUMBER	0	3	2	Zone risk as computed by DSS.
probabilityOfAttackLatent	NUMBER	0	3	2	The original "baseline" poa value.
OPRiskC	NUMBER		6	2	Operational risk to confidentiality.
OPRiskI	NUMBER		6	2	Operational Risk to Integrity
OPRiskA	NUMBER		6	2	Operational risk to Availability.
notes	VARCHAR2	2000			Notes about the zone. May be entered by the system or by the operator.

Connections

Connector	Source	Target
<u>Association</u> (zoneID = zoneID) Source -> Destination	Topology:ZoneSubnet	Topology:Zone
<u>Association</u> (zoneID = zoneID) Source -> Destination	Topology:ZoneAsset	Topology:Zone
<u>Association</u> (ZoneID = zoneID) Source -> Destination	RFC:RFCZone	Topology:Zone
<u>Association</u> (ZoneID = zoneID) Source -> Destination	Topology:ZoneLocations	Topology:Zone
<u>Association</u> (ZoneID = zoneID) Source -> Destination	Point of Contact:ZonePOC	Topology:Zone
<u>Association</u> (zoneID = zoneID) Source -> Destination	Topology:ZoneBorder	Topology:Zone
<u>Dependency</u> calls Source -> Destination	Topology:Zone	Topology:ZoneIDSeq

Operations

Method	Parameter
PK_Zone()	<u>NUMBER</u> zoneID [in]
UQ_Zone_label()	<u>VARCHAR2</u> label [in]
TRG_PreInsert_Zone before insert()	

3.5.4.3.3.3 ZoneAsset

Type: **Class**

Stereotype: *«table»*

Notes:

Optional link when the IP Address is not going to be available. Normally assets link to zones via subnets and their IP addresses.

Attributes

Name	Type	Length	Precision	Scale	Notes
zoneID	NUMBER		16	0	Foreign key to zone table
serviceID	NUMBER		12	0	Foreign key to asset table
Count	NUMBER		12	0	The number of assets of this type in the zone

Connections

Connector	Source	Target
<u>Association</u> (zoneID = zoneID) Source -> Destination	Topology:ZoneAsset	Topology:Zone
<u>Association</u> (serviceID = serviceID) Source -> Destination	Topology:ZoneAsset	Asset:Asset

Operations

Method	Parameter
FK_ZoneAsset_Asset()	<u>NUMBER</u> serviceID [in]
FK_ZoneAsset_Zone()	<u>NUMBER</u> zoneID [in]
PK_ZoneAsset()	<u>NUMBER</u> zoneID [in] <u>NUMBER</u> serviceID [in]

3.5.4.3.3.4 ZoneBorder

Type: **Class**

Stereotype: «*table*»

Notes:

The zone border table stores the assets that represent the perimeter of a zone. The asset identified must be the asset that can enforce the rules that create the zone.

Attributes

Name	Type	Length	Precision	Scale	Notes
serviceID	NUMBER	0	12	0	The asset service ID of the network interface that creates a border to the zone.
zoneID	NUMBER	0	16	0	The Zone ID that the border asset defines.
created	TIMESTAMP	0	0	0	Timestamp of record creation.
modified	TIMESTAMP	0	0	0	Timestamp of last modification

Connections

Connector	Source	Target
<u>Association</u> (zoneID = serviceID zoneServiceID = zoneID)(zoneID = zoneID zoneServiceID = serviceID) Source -> Destination	Topology:ZoneBorderSafeguard	Topology:ZoneBorder
<u>Association</u> (zoneID = zoneID) Source -> Destination	Topology:ZoneBorder	Topology:Zone
<u>Association</u> (zoneID = zoneID serviceID = serviceID) Source -> Destination	Topology:ZoneRule	Topology:ZoneBorder
<u>Association</u> (serviceID = serviceID) Source -> Destination	Topology:ZoneBorder	Asset:Asset

Operations

Method	Parameter
FK_ZoneBorder_Asset()	<u>NUMBER</u> serviceID [in]
FK_ZoneBorder_Zone()	<u>NUMBER</u> zoneID [in]
PK_ZoneBorder()	<u>NUMBER</u> zoneID [in] <u>NUMBER</u> serviceID [in]
TRG_PreInsert_ZoneBorder before insert()	

3.5.4.3.3.5 ZoneBorderSafeguard

Type: **Class**

Stereotype: «table»

Notes:

This table stores the perimeter safeguards between zones.

Attributes

Name	Type	Length	Precision	Scale	Notes
zoneID	NUMBER		16	0	The zoneID for the zone border
zoneServiceID	NUMBER		12	0	The serviceID to identify the zone border
safeguardID	VARCHAR2	50			The safeguardID of the perimeter safeguard
safeguardServiceID	NUMBER		12	0	The serviceID of the safeguard Implementation

Connections

Connector	Source	Target
<u>Association</u> (zoneID = serviceID zoneServiceID = zoneID)(zoneID = zoneID zoneServiceID = serviceID) Source -> Destination	Topology:ZoneBorderSafeguard	Topology:ZoneBorder
<u>Association</u> (safeguardID = safeguardID safeguardServiceID = serviceID) Source -> Destination	Topology:ZoneBorderSafeguard	Safeguard:SafeguardImplementation

Operations

Method	Parameter
FK_ZoneBorderSaf_SafeguardImpl()	<u>VARCHAR2</u> safeguardID [in] <u>NUMBER</u> safeguardServiceID [in]
FK_ZoneBorderSafegu_ZoneBorder()	<u>NUMBER</u> zoneID [in] <u>NUMBER</u> zoneServiceID [in]
PK_ZoneBorderSafeguard()	<u>NUMBER</u> zoneID [in] <u>NUMBER</u> zoneServiceID [in] <u>VARCHAR2</u> safeguardID [in]

Method	Parameter
	<u>NUMBER</u> safeguardServiceID [in]

3.5.4.3.3.6 ZoneIDSeq

Type: Class

Stereotype: «db_sequence»

Notes:

Connections

Connector	Source	Target
<u>Dependency</u> calls Source -> Destination	Topology:Zone	Topology:ZoneIDSeq

3.5.4.3.3.7 ZoneLocations

Type: **Class**

Stereotype: «table»

Notes:

Optional list of locations that a zone is comprised of.

Attributes

Name	Type	Length	Precision	Scale	Notes
ZoneID	NUMBER		16	0	Foreign key to the Zone Table
LocationName	VARCHAR2	200			Foreign key to the Location table

Connections

Connector	Source	Target
<u>Association</u> (LocationName = LocationName) Source -> Destination	Topology:ZoneLocations	Location:Location
<u>Association</u> (ZoneID = zoneID) Source -> Destination	Topology:ZoneLocations	Topology:Zone

Operations

Method	Parameter
FK_ZoneLocations_Location()	<u>VARCHAR2</u> LocationName [in]
FK_ZoneLocations_Zone()	<u>NUMBER</u> ZoneID [in]
PK_ZoneLocations()	<u>NUMBER</u> ZoneID [in] <u>VARCHAR2</u> LocationName [in]

3.5.4.3.3.8 ZoneRule

Type: **Class**

Stereotype: «table»

Notes:

The list of rules being enforced for the zone.

Attributes

Name	Type	Length	Precision	Scale	Notes
serviceID	NUMBER	0	12	0	Asset that is providing the "border" to the zone.
zoneID	NUMBER	0	16	0	The zone ID from the parent.
ruleID	NUMBER	0	6	0	RuleID is a sequence within parent but should not be used for ordering the rules. Order is a separate column since order may change it is not part of the PK.
ruleOrder	NUMBER	0	6	0	Numeric ordering since the order in which rules are encountered is important.
created	TIMESTAMP	0	0	0	Timestamp of record creation
modified	TIMESTAMP	0	0	0	Timestamp of last modification.
permit	CHAR	1	0	0	'Y' OR 'N'
fromsubnetID	NUMBER	0	12	0	The JNDMS unique subnet identifier used to define absolute addresses.
fromIP	VARCHAR2	64	0	0	IP of the from subnet associated with the rule.
startFromIP	NUMBER	0	38	0	The decimal IP of the starting point of the subnet.
endFromIP	NUMBER	0	38	0	The decimal IP of the end point of the subnet.
fromPort	NUMBER	0	6	0	Port number from the originator.
toSubnetID	NUMBER	0	12	0	The unique JNDMS subnet id.
toIP	VARCHAR2	64	0	0	IP address of the to subnet.
startToIP	NUMBER	0	38	0	The decimal IP of the starting point of the subnet.

DN0678
Issue 4/2: 06 February 2012

Name	Type	Length	Precision	Scale	Notes
endToIP	NUMBER	0	38	0	The decimal IP of the end point of the subnet.
toPort	NUMBER	0	6	0	To port number of the rule.
protocol	VARCHAR2	50	0	0	Packet protocol of the rule.
ruleText	VARCHAR2	2000	0	0	A copy of the actual rule text provided by the source for easy review by the analyst.
transformation	VARCHAR2	200			Lists any transformations that are performed on the data as it passes through the zone with this rule. Examples include encryption, decryption and NAT.

Connections

Connector	Source	Target
<u>Association</u> (zoneID = zoneID serviceID = serviceID) Source -> Destination	Topology:ZoneRule	Topology:ZoneBorder

Operations

Method	Parameter
FK_ZoneRule_ZoneBorder()	<u>NUMBER</u> zoneID [in] <u>NUMBER</u> serviceID [in]
PK_rules()	<u>NUMBER</u> zoneID [in] <u>NUMBER</u> ruleID [in] <u>NUMBER</u> serviceID [in]
TRG_PreInsert_ZoneRule before insert()	
TRG_IPUpdate_ZoneRule before insert or update()	

3.5.4.3.3.9 ZoneSubnet

Type: **Class**

Stereotype: «table»

Notes:

The list of subnets for a given zone.

Attributes

Name	Type	Length	Precision	Scale	Notes
zoneID	NUMBER	0	16	0	PK from parent.
zoneSubnetID	NUMBER	0	12	0	Sequence within parent of the Zone.
IP	VARCHAR2	64	0	0	The relative IP address of the subnet in decimal form,
created	TIMESTAMP	0	0	0	Timestamp of record creation.
modified	TIMESTAMP	0	0	0	Timestamp of last modification.
startIP	NUMBER	0	38	0	The decimal representation of the starting IP.
endIP	NUMBER	0	38	0	End IP of the subnet in decimal format.

Connections

Connector	Source	Target
<u>Association</u> (zoneID = zoneID) Source -> Destination	Topology:ZoneSubnet	Topology:Zone
<u>Association</u> Source -> Destination	Topology:ZoneSubnet	Asset:Asset

Operations

Method	Parameter
FK_ZoneSubnet_Zone()	<u>NUMBER</u> zoneID [in]
PK_ZoneSubnet()	<u>NUMBER</u> zoneID [in] <u>NUMBER</u> zoneSubnetID [in]
TRG_PreInsert_ZoneSubnet before insert()	
TRG_IPUpdate_ZoneSubnet before insert or update()	

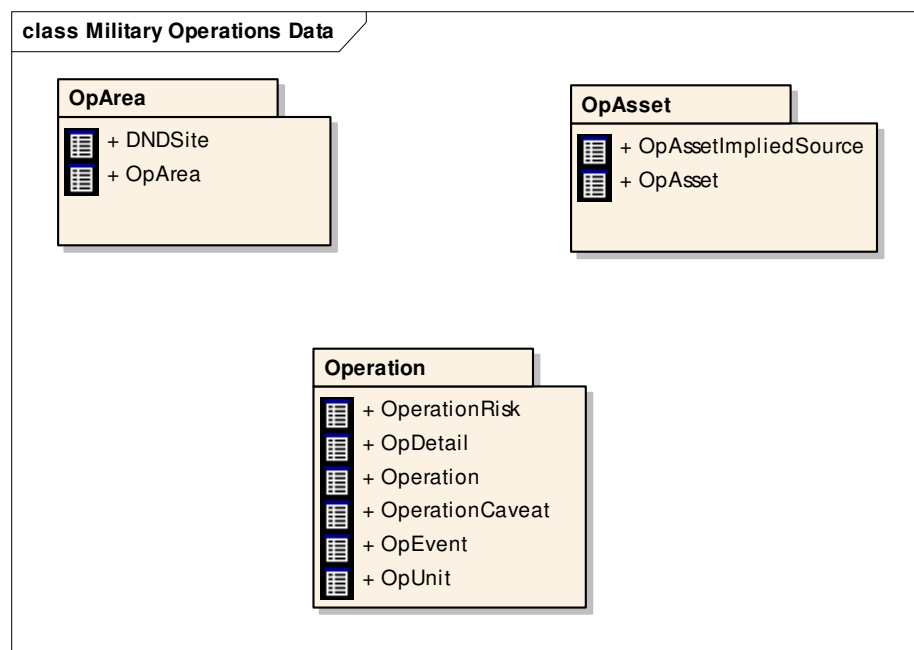
3.5.4.4 Military Operations Data

Operational Data reflects the operations being performed by the organization. The operational data will be correlated to the IT Infrastructure to record the operation requirements and for use in assessing impact to operations. The intent of JNDMS is to be able to link and collect operational data from external systems; however, this will be difficult as the correlation to the IT Infrastructure is not supported by any known tool. In reality, each and every C4ISR system will implement its own data model; however some standards do exist. The JC3IEDM/C2IEDM are NATO standard data models that have been developed to identify a common data structure to be used for command and control (C2) data exchange. The C2 domain is vast and conceivably could encompass the other two model types previously presented. For the purposes of the JNDMS, the focus will be on the operational data and linking it to the other data models previously presented.

Military Operations Data will be captured into the database with these inputs:

1. Operations Name
2. Operation Locations
3. Operations IT Assets
4. Operation IT Asset Policy
5. Operation Priority
6. Operation Activity Schedule
7. Operation Organizations

Military Operations Data - (Logical diagram)

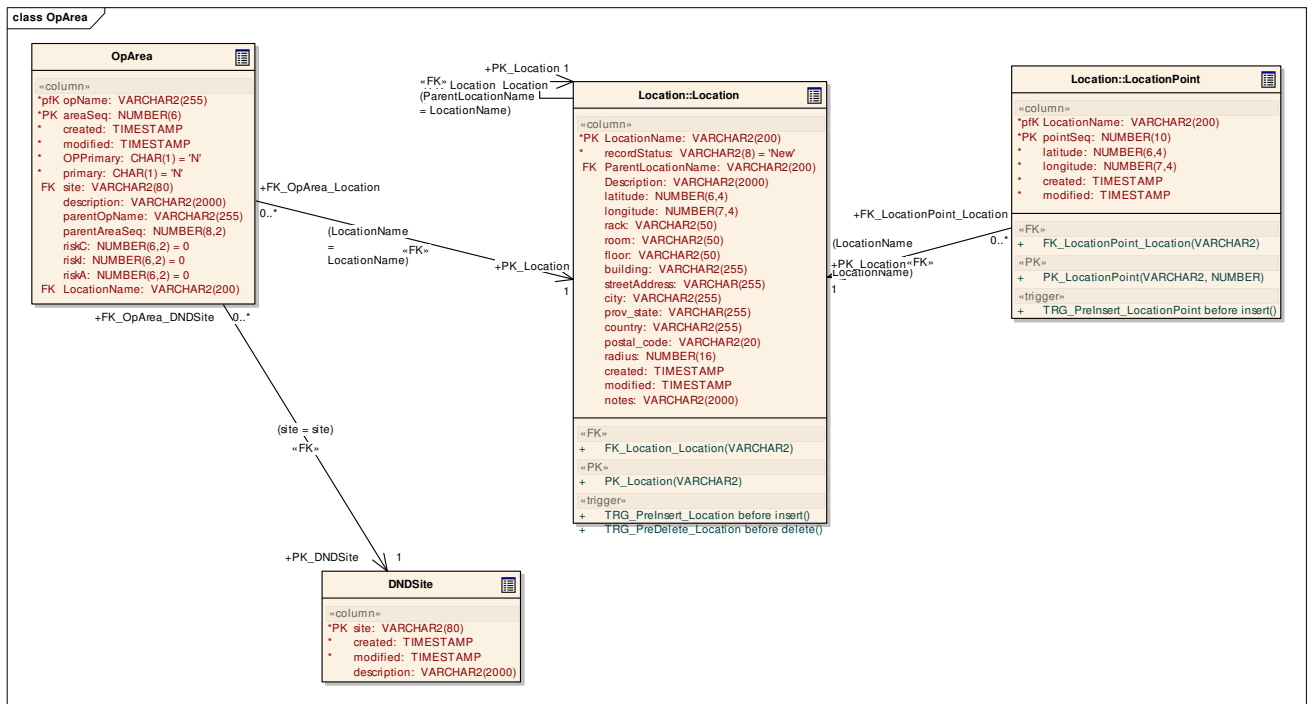


3.5.4.4.1 OpArea

The OpArea package contains the entities related to Operation Area.

The Op Area is the only reference to a location within the data model. By assuming that even day-to-day support of CF operations is an operation, each and every asset can be given a location based upon its operation. A simpler model of giving assets a direct location value was considered. The problem with the simpler model is when modeling asset location and operation area, it is at the very least redundant data for any asset that is associated with an operation. The simple model even allows for the data to be inconsistent between operation area and asset location. As modeled, these problems are eliminated since there is only one possible location value for an asset.

OpArea - (Logical diagram)



3.5.4.4.2 DNDSite

Type: **Class**

Stereotype: «table»

Notes:

A list of DND Sites (i.e., bases, ships, deployed HQs) to support filtering.

Attributes

Name	Type	Length	Precision	Scale	Notes
site	VARCHAR2	80	0	0	Site name.
created	TIMESTAMP	0	0	0	Timestamp record was created.
modified	TIMESTAMP	0	0	0	timestamp of last modification.
description	VARCHAR2	2000	0	0	A detailed free text description of the site.

Connections

Connector	Source	Target
<u>Association</u> (site = site) Source -> Destination	OpArea:OpArea	OpArea:DNDSite

Operations

Method	Parameter
PK_DNDSite()	<u>VARCHAR2</u> site [in]
TRG_PreInsert_DNDSite before insert()	

3.5.4.4.3 OpArea

Type: **Class**

Stereotype: «table»

Notes:

The OpArea may be either a location or a closed polygon as indicated in the type field. If using the closed polygon, details must be gathered from the OpAreaPoint table.

This table allows for multiple simple closed polygons to be used to represent the Operation Area. For JNDMS, the operation area is not critical. It is intended only to be used to provide the commander a reference to the operation area on a map/GIS display.

One operation area will be designated as the primary area. If asset locations are assigned to an Operation, then the asset will show in the primary area.

Attributes

Name	Type	Length	Precision	Scale	Notes
opName	VARCHAR2	255	0	0	The operation name.
areaSeq	NUMBER	0	6	0	Location sequence number, supporting multiple operation areas for a single operation.
created	TIMESTAMP	0	0	0	Timestamp when the record was created.
modified	TIMESTAMP	0	0	0	Timestamp when the record was last modified.
OPPrimary	CHAR	1	0	0	Flag ('Y' or 'N') indicates whether or not this OpArea is the primary area for the operation.
primary	CHAR	1	0	0	Flag ('Y' or 'N') indicate if this is the primary op area for the DND site. A trigger should be defined to verify that only 1 operation area per DND SITE is defined as primary.
site	VARCHAR2	80			The DND site.
description	VARCHAR2	2000	0	0	Text description of the operation area.
parentOpName	VARCHAR2	255			The name of the hierarchical parent of this OpArea.
parentAreaSeq	NUMBER		8	2	The area sequence of the hierarchical parent of this OpArea.

DN0678
Issue 4/2: 06 February 2012

Name	Type	Length	Precision	Scale	Notes
riskC	NUMBER		6	2	Intermediate risk value used by the DSS (Confidentiality)
riskI	NUMBER		6	2	Intermediate risk value used by the DSS (Integrity)
riskA	NUMBER		6	2	Intermediate risk value used by the DSS (Availability)
LocationName	VARCHAR2	200			The name of the location

Connections

Connector	Source	Target
<u>Association</u> (opName = name) Source -> Destination	OpArea:OpArea	Operation:Operation
<u>Association</u> (opName = opName areaSeq = areaSeq) Source -> Destination	Operation:OpDetail	OpArea:OpArea
<u>Association</u> (site = site) Source -> Destination	OpArea:OpArea	OpArea:DNDSite
<u>Association</u> (LocationName = LocationName) Source -> Destination	OpArea:OpArea	Location:Location
<u>AssociationClass</u> Source -> Destination	OpArea:OpArea	C2IEDM:ACT_LOC

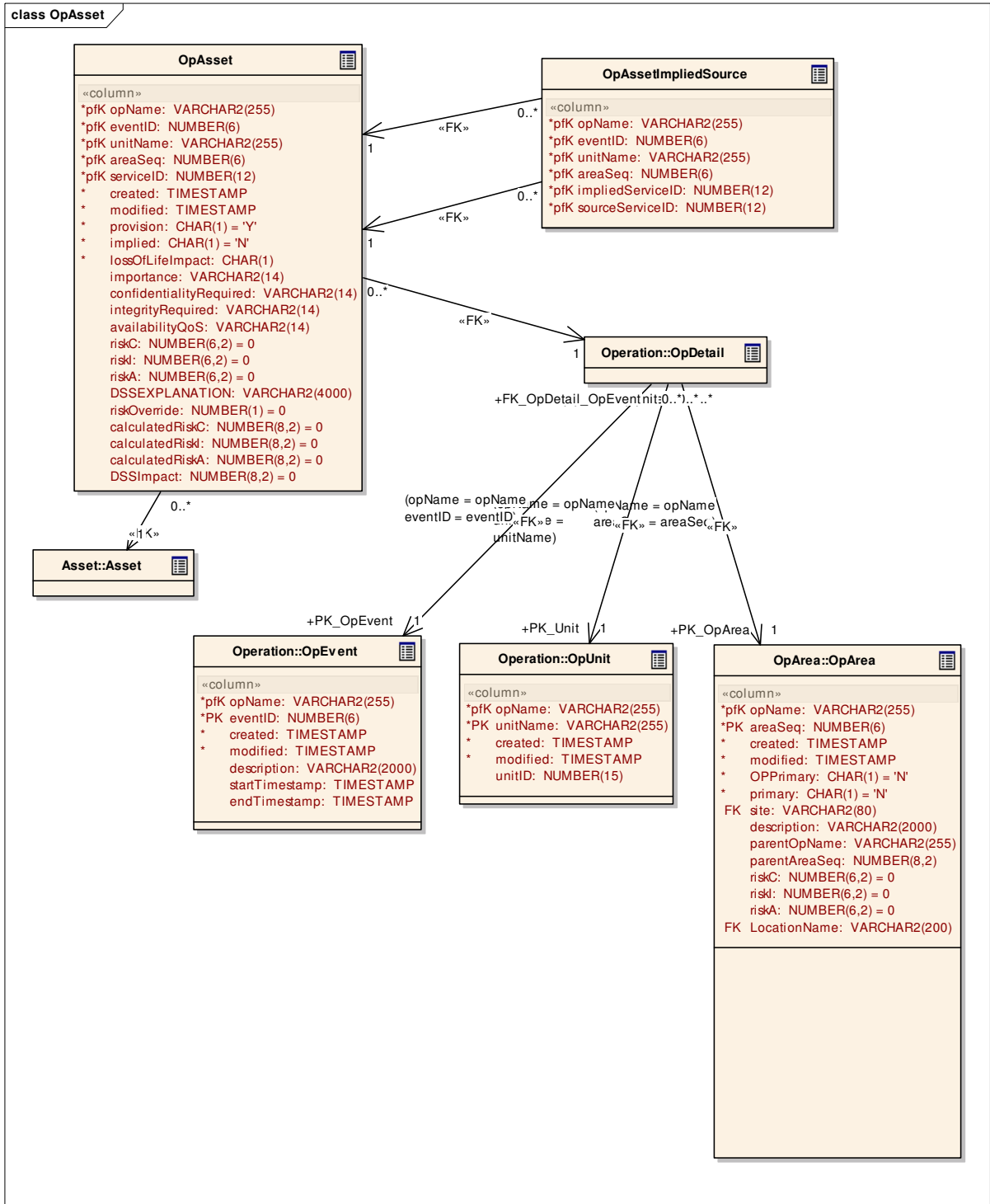
Operations

Method	Parameter
FK_OpArea_Location()	<u>VARCHAR2</u> LocationName [in]
FK_OpArea_DNDSite()	<u>VARCHAR2</u> site [in]
PK_OpArea()	<u>VARCHAR2</u> opName [in] <u>NUMBER</u> areaSeq [in]
FK_OpArea_Operation()	<u>VARCHAR2</u> opName [in]
TRG_PreInsert_OpArea before insert()	
TRG_PreUpdate_oparea before update of opname()	

3.5.4.4.4 OpAsset

The OpAsset package contains entities that describe operations and shows the relationships used to map operations to assets.

OpAsset - (Logical diagram)



3.5.4.4.4.1 OpAsset

Type: **Class**

Stereotype: «table»

Notes:

The operation asset is the critical link between CND data and the Operations. It records the assets required to support each operation. It is an intersection table as a single operation can require many assets and an asset may be required by many operations. Several additional attributes that are captured may be used for risk and or impact calculations.

Attributes

Name	Type	Length	Precision	Scale	Notes
opName	VARCHAR2	255	0	0	Operation name requiring the asset.
eventID	NUMBER	0	6	0	The operational event id.
unitName	VARCHAR2	255	0	0	Operational unit asset is to be assigned to.
areaSeq	NUMBER	0	6	0	Operation area of the unit for the event.
serviceID	NUMBER	0	12	0	The asset service ID required my the mission.
created	TIMESTAMP	0	0	0	Timestamp when record was created.
modified	TIMESTAMP	0	0	0	Timestamp of last modification to the record.
provision	CHAR	1			Is the service/asset to be provided by the operation or only accessed.
implied	CHAR	1			Indicates if the asset assignment to the op was explicit or implied by selection of an asset (service) that is hosted by this asset.
lossOfLifeImpact	CHAR	1	0	0	Y or N. Is Loss of Life a potential risk/impact of losing service of this asset.

DN0678
Issue 4/2: 06 February 2012

Name	Type	Length	Precision	Scale	Notes
importance	VARCHAR2	14	0	0	This is the importance of this service to the mission. This is used to plan graceful degradation of service. Values include essential, very important, important, useful and not applicable.
confidentialityRequired	VARCHAR2	14	0	0	Confidentiality required of this asset to support the mission. Values include critical, high, medium, low or not applicable.
integrityRequired	VARCHAR2	14	0	0	Integrity required of this asset to support the mission. Values include critical, high, medium, low or not applicable.
availabilityQoS	VARCHAR2	14	0	0	Availability required of this asset to support the mission. Values include critical, high, medium, low or not applicable.
riskC	NUMBER		6	2	Intermediate risk value used by the DSS (Confidentiality)
riskI	NUMBER		6	2	Intermediate risk value used by the DSS (Integrity)
riskA	NUMBER		6	2	Intermediate risk value used by the DSS (Availability)
DSSEXPLANATION	VARCHAR2	4000			An explanation from the DSS of its calculations
riskOverride	NUMBER		1	0	allow the user to override the risk calculation to keep their own risk numbers in the risk c,i,a columns
calculatedRiskC	NUMBER		8	2	calculated Risk from the DSS
calculatedRiskI	NUMBER		8	2	calculated Risk from the DSS
calculatedRiskA	NUMBER		8	2	calculated Risk from the DSS
DSSImpact	NUMBER		8	2	The impact of an event on an operation because of an asset

Connections

Connector	Source	Target
<p><u>Association</u>(opName = opName eventID = eventID unitName = unitName areaSeq = areaSeq) Source -> Destination</p>	OpAsset:OpAsset	Operation:OpDetail
<p><u>Association</u>(opName = opName unitName = eventID eventID = unitName areaSeq = areaSeq sourceServiceID = serviceID)(opName = opName eventID = eventID unitName = unitName areaSeq = areaSeq sourceServiceID = serviceID) Source -> Destination</p>	OpAsset:OpAssetImpliedSource	OpAsset:OpAsset
<p><u>Association</u>(opName = opName eventID = eventID unitName = unitName areaSeq = areaSeq impliedServiceID = serviceID) Source -> Destination</p>	OpAsset:OpAssetImpliedSource	OpAsset:OpAsset
<p><u>Association</u>(serviceID = serviceID) Source -> Destination</p>	OpAsset:OpAsset	Asset:Asset

Operations

Method	Parameter
FK_OpAsset_Asset()	<u>NUMBER</u> serviceID [in]
FK_OpAsset_OpDetail()	<u>VARCHAR2</u> opName [in] <u>NUMBER</u> eventID [in] <u>VARCHAR2</u> unitName [in] <u>NUMBER</u> areaSeq [in]
PK_OpAsset()	<u>VARCHAR2</u> opName [in] <u>NUMBER</u> eventID [in] <u>VARCHAR2</u> unitName [in] <u>NUMBER</u> areaSeq [in] <u>NUMBER</u> serviceID [in]
TRG_PreInsert_OpAsset before insert()	
CHK_provision()	
CHK_implied()	
TRG_PreUpdate_OpAssetRiskOver before update of riskOverride()	
TRG_PreUpdate_OpAssetRiskC before update of riskc()	
TRG_PreUpdate_OpAssetRiskI before update of riski()	
TRG_PreUpdate_OpAssetRiskA before update of riskA()	

3.5.4.4.2 OpAssetImpliedSource

Type: **Class**

Stereotype: «table»

Notes:

The Implied Source links the OpAsset table to the OpAsset table. It indicates the source of an asset that was added. This is relevant when an asset is only included as an operational asset when the reason it is added is due to another asset. Software installed on an asset would be one example of an OpAsset with an implied source as the host machine. The assets always share the same OpEvent/OpArea/OpUnit entries.

Attributes

Name	Type	Length	Precision	Scale	Notes
opName	VARCHAR2	255			Foreign key to the OpAsset table
eventID	NUMBER		6	0	Foreign key to the OpAsset table
unitName	VARCHAR2	255			Foreign key to the OpAsset table
areaSeq	NUMBER		6	0	Foreign key to the OpAsset table
impliedServiceID	NUMBER		12	0	Foreign key to the OpAsset table
sourceServiceID	NUMBER		12	0	Foreign key to the OpAsset table

Connections

Connector	Source	Target
Association (opName = opName unitName = eventID eventID = unitName areaSeq = areaSeq sourceServiceID = serviceID)(opName = opName eventID = eventID unitName = unitName areaSeq = areaSeq sourceServiceID = serviceID) Source -> Destination	OpAsset:OpAssetImpliedSource	OpAsset:OpAsset

Connector	Source	Target
Association (opName = opName eventID = eventID unitName = unitName areaSeq = areaSeq impliedServiceID = serviceID) Source -> Destination	OpAsset:OpAssetImpliedSource	OpAsset:OpAsset

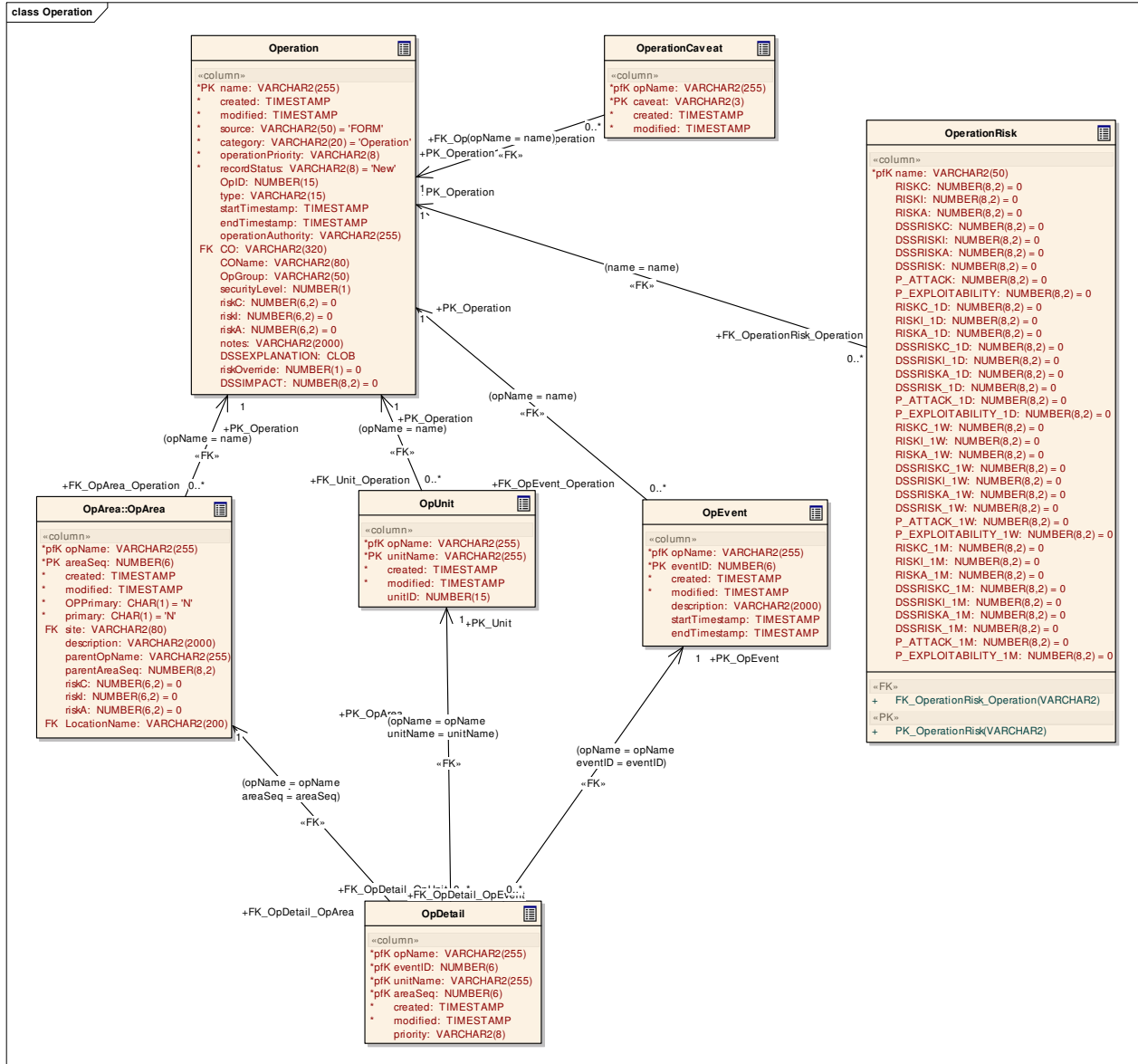
Operations

Method	Parameter
FK_OpAssetImplSrc_OpAsset()	<u>VARCHAR2</u> opName [in] <u>NUMBER</u> eventID [in] <u>VARCHAR2</u> unitName [in] <u>NUMBER</u> areaSeq [in] <u>NUMBER</u> impliedServiceID [in]
FK_OpAssetImplSrc_OpAssetSrc()	<u>VARCHAR2</u> opName [in] <u>NUMBER</u> eventID [in] <u>VARCHAR2</u> unitName [in] <u>NUMBER</u> areaSeq [in] <u>NUMBER</u> sourceServiceID [in]
PK_OpAssetImpliedSource()	<u>VARCHAR2</u> opName [in] <u>NUMBER</u> eventID [in] <u>VARCHAR2</u> unitName [in] <u>NUMBER</u> areaSeq [in] <u>NUMBER</u> impliedServiceID [in] <u>NUMBER</u> sourceServiceID [in]

3.5.4.4.5 Operation

The Operation package contains the entities that describe the operation and the detailed portions of the operation. This data will be collected from an operational database (via the C2IEDM) or via the UI.

Operation - (Logical diagram)



3.5.4.4.5.1 OpDetail

Type: **Class**

Stereotype: «table»

Notes:

The Operation detail captures the combination of a specific event, unit, area and asset. This will show the detailed requirement and location of the asset.

Attributes

Name	Type	Length	Precision	Scale	Notes
opName	VARCHAR2	255	0	0	Operation name.
eventID	NUMBER	0	6	0	Operation event.
unitName	VARCHAR2	255	0	0	Unit involved in operational event.
areaSeq	NUMBER	0	6	0	Operation area associated with the unit and the event.
created	TIMESTAMP	0	0	0	Timestamp of record creation.
modified	TIMESTAMP	0	0	0	Timestamp of last modification.
priority	VARCHAR2	8	0	0	Critical High Medium Low N/A

Connections

Connector	Source	Target
Association (opName = opName eventID = eventID) Source -> Destination	Operation:OpDetail	Operation:OpEvent
Association (opName = opName areaSeq = areaSeq) Source -> Destination	Operation:OpDetail	OpArea:OpArea

Connector	Source	Target
Association (opName = opName eventID = eventID unitName = unitName areaSeq = areaSeq) Source -> Destination	OpAsset:OpAsset	Operation:OpDetail
Association (opName = opName unitName = unitName) Source -> Destination	Operation:OpDetail	Operation:OpUnit

Operations

Method	Parameter
FK_OpDetail_OpArea()	<u>VARCHAR2</u> opName [in] <u>NUMBER</u> areaSeq [in]
FK_OpDetail_OpEvent()	<u>VARCHAR2</u> opName [in] <u>NUMBER</u> eventID [in]
FK_OpDetail_OpUnit()	<u>VARCHAR2</u> opName [in] <u>VARCHAR2</u> unitName [in]
PK_OpDetail()	<u>VARCHAR2</u> opName [in] <u>NUMBER</u> eventID [in] <u>VARCHAR2</u> unitName [in] <u>NUMBER</u> areaSeq [in]
TRG_PreInsert_OpDetail before insert()	
TRG_PreUpdate_OpDetail before update of opname()	

3.5.4.4.5.2 Operation

Type: **Class**

Stereotype: «table»

Notes:

The Operation represents all operations carried out by the DND. This ranges from day-to-day base operations to deployed international operations.

Attributes

Name	Type	Length	Precision	Scale	Notes
name	VARCHAR2	255	0	0	The operation name will be the Primary Key of the operation. This benefit is that the operation name has enough meaning that in most database queries, a look-up of the operation data will not be required as the name is enough detail to the operator to put the query result in context.
created	TIMESTAMP	0	0	0	A timestamp recorded when the record is originally created in the database.
modified	TIMESTAMP	0	0	0	A timestamp recorded in the database every time the record is changed.
source	VARCHAR2	50	0	0	Source indicates the source of the operational data. The data may come from a JNDMS GUI (form) or from some other system (i.e.. a C2IEDM data exchange).
category	VARCHAR2	20			Operation category. Examples include Training, Exercise, Operation and Planning.
operationPriority	VARCHAR2	8	0	0	Critical High Medium Routine Exercise N/A

DN0678
Issue 4/2: 06 February 2012

Name	Type	Length	Precision	Scale	Notes
recordStatus	VARCHAR2	8			Indicates the jndms internal status of the data. Allowable values include: New - A new record in the DB. Pending - A record in the DB that is being reviewed. Approved - A record that has been reviewed and approved for use. Denied - a record that is not to be used by the JNDMS but is retained in the DB.
OpID	NUMBER	0	15	0	This column maintains the primary key for other systems (IAT/C2IEDM) but will not be used as the primary key in JNDMS for performance reasons.
type	VARCHAR2	15	0	0	International Deployed Domestic
startTimestamp	TIMESTAMP	0	0	0	Start time of the operation.
endTimestamp	TIMESTAMP	0	0	0	End time of the operation.
operationAuthority	VARCHAR2	255			The country or organization that has authority over the operation or leads the operation.
CO	VARCHAR2	320	0	0	Commanding Officer There is an assumption here that the CO is a JNDMS user.
COName	VARCHAR2	80	0	0	In the event the CO is not a JNDMS user, this field will allow operators to record the CO for the operation.
OpGroup	VARCHAR2	50	0	0	Organization responsible for maintaining the operations data.

DN0678
Issue 4/2: 06 February 2012

Name	Type	Length	Precision	Scale	Notes
securityLevel	NUMBER	0	1	0	The security classification of the operation.
riskC	NUMBER		6	2	Intermediate risk value used by the DSS (Confidentiality)
riskI	NUMBER		6	2	Intermediate risk value used by the DSS (Integrity)
riskA	NUMBER		6	2	Intermediate risk value used by the DSS (Availability)
notes	VARCHAR2	2000			Notes on the record. May be entered by operator or automatically.
DSSEXPLANATION	CLOB				An explanation for the DSS's calculations.
riskOverride	NUMBER		1	0	Set to 1 if the risk values have been overridden by the user
DSSIMPACT	NUMBER		8	2	An indication of how it has been impacted by its assets' statuses.

Connections

Connector	Source	Target
<u>Association</u> (opName = name) Source -> Destination	OpArea:OpArea	Operation:Operation
<u>Association</u> (CO = ID) Source -> Destination	Operation:Operation	Users:UserProfile
<u>Association</u> (opName = name) Source -> Destination	Operation:OpEvent	Operation:Operation
<u>Association</u> (opName = name) Source -> Destination	Operation:OpUnit	Operation:Operation
<u>Association</u> (opName = name) Source -> Destination	Operation:OperationCaveat	Operation:Operation

Connector	Source	Target
<u>Association</u> (name = name) Source -> Destination	Operation:OperationRisk	Operation:Operation
<u>Association</u> (OpName = name) Source -> Destination	Point of Contact:OperationPOC	Operation:Operation
<u>AssociationClass</u> Source -> Destination	Operation:Operation	C2IEDM:ACT_TASK
<u>AssociationClass</u> Source -> Destination	J6:J6Operations	Operation:Operation
<u>AssociationClass</u> Unspecified	Operation:Operation	J6 BE:J6Operations
<u>AssociationClass</u> Source -> Destination	Operation:Operation	C2IEDM:ACT

Operations

Method	Parameter
FK_CO()	<u>VARCHAR2</u> CO [in]
PK_Operation()	<u>VARCHAR2</u> name [in]
TRG_PreInsert_Operation before insert()	
TRG_PreUpdate_operation before update of name()	
CHK_OPERATION_CATEGORY()	

3.5.4.4.5.3 OperationCaveat

Type: Class

Stereotype: «table»

Notes:

This table contains the list of applicable countries (caveats) associated with the operation data.

Attributes

Name	Type	Length	Precision	Scale	Notes
opName	VARCHAR2	255	0	0	The operation name.
caveat	VARCHAR2	3	0	0	The country code (caveat).
created	TIMESTAMP	0	0	0	Timestamp record was created.
modified	TIMESTAMP	0	0	0	Timestamp of last modification.

Connections

Connector	Source	Target
<u>Association</u> (opName = name) Source -> Destination	Operation:OperationCaveat	Operation:Operation

Operations

Method	Parameter
FK_OperationCaveats_Operation()	<u>VARCHAR2</u> opName [in]
PK_OperationCaveats()	<u>VARCHAR2</u> opName [in] <u>VARCHAR2</u> caveat [in]
TRG_PreInsert_OperationCaveat before insert()	

3.5.4.4.5.4 OperationRisk

Type: **Class**

Stereotype: «table»

Notes:

A table for storing the DSS working values for calculating DSS risk

Attributes

Name	Type	Length	Precision	Scale	Notes
name	VARCHAR2	50			Operation's name
RISKC	NUMBER		8	2	Intermediate risk value used by the DSS
RISKI	NUMBER		8	2	Intermediate risk value used by the DSS
RISKA	NUMBER		8	2	Intermediate risk value used by the DSS
DSSRISKC	NUMBER		8	2	Intermediate risk value used by the DSS
DSSRISKI	NUMBER		8	2	Intermediate risk value used by the DSS
DSSRISKA	NUMBER		8	2	Intermediate risk value used by the DSS
DSSRISK	NUMBER		8	2	Intermediate risk value used by the DSS
P_ATTACK	NUMBER		8	2	Probability of attack
P_EXPLOITABILITY	NUMBER		8	2	Probability of exploitation
RISKC_1D	NUMBER		8	2	Intermediate risk value used by the DSS (estimated 1 day in the future)
RISKI_1D	NUMBER		8	2	Intermediate risk value used by the DSS (estimated 1 day in the future)
RISKA_1D	NUMBER		8	2	Intermediate risk value used by the DSS (estimated 1 day in the future)
DSSRISKC_1D	NUMBER		8	2	Intermediate risk value used by the DSS (estimated 1 day in the future)
DSSRISKI_1D	NUMBER		8	2	Intermediate risk value used by the DSS (estimated 1 day in the future)

DN0678
Issue 4/2: 06 February 2012

Name	Type	Length	Precision	Scale	Notes
DSSRISKA_1D	NUMBER		8	2	Intermediate risk value used by the DSS (estimated 1 day in the future)
DSSRISK_1D	NUMBER		8	2	Intermediate risk value used by the DSS (estimated 1 day in the future)
P_ATTACK_1D	NUMBER		8	2	Probability of Attack (Estimated 1d in the future)
P_EXPLOITABILITY_1D	NUMBER		8	2	Probability of Exploitation (estimated 1 day in the future)
RISKC_1W	NUMBER		8	2	Intermediate risk value used by the DSS (estimated 1 week in the future)
RISKI_1W	NUMBER		8	2	Intermediate risk value used by the DSS (estimated 1 week in the future)
RISKA_1W	NUMBER		8	2	Intermediate risk value used by the DSS (estimated 1 week in the future)
DSSRISKC_1W	NUMBER		8	2	Intermediate risk value used by the DSS (estimated 1 week in the future)
DSSRISKI_1W	NUMBER		8	2	Intermediate risk value used by the DSS (estimated 1 week in the future)
DSSRISKA_1W	NUMBER		8	2	Intermediate risk value used by the DSS (estimated 1 week in the future)
DSSRISK_1W	NUMBER		8	2	Intermediate risk value used by the DSS (estimated 1 week in the future)
P_ATTACK_1W	NUMBER		8	2	Probability of attack (estimated 1 week in the future)
P_EXPLOITABILITY_1W	NUMBER		8	2	Probability of Exploitation (estimated 1 week in the future)
RISKC_1M	NUMBER		8	2	Intermediate risk value used by the DSS (estimated 1 month in the future)
RISKI_1M	NUMBER		8	2	Intermediate risk value used by the DSS (estimated 1 month in the future)

DN0678
Issue 4/2: 06 February 2012

Name	Type	Length	Precision	Scale	Notes
RISKA_1M	NUMBER		8	2	Intermediate risk value used by the DSS (estimated 1 month in the future)
DSSRISKC_1M	NUMBER		8	2	Intermediate risk value used by the DSS (estimated 1 month in the future)
DSSRISKI_1M	NUMBER		8	2	Intermediate risk value used by the DSS (estimated 1 month in the future)
DSSRISKA_1M	NUMBER		8	2	Intermediate risk value used by the DSS (estimated 1 month in the future)
DSSRISK_1M	NUMBER		8	2	Intermediate risk value used by the DSS (estimated 1 month in the future)
P_ATTACK_1M	NUMBER		8	2	Probability of Attack (estimated 1 month in the future)
P_EXPLOITABILITY_1M	NUMBER		8	2	Probability of Exploitation (estimated 1 month in the future)

Connections

Connector	Source	Target
<u>Association</u> (name = name) Source -> Destination	Operation:OperationRisk	Operation:Operation

Operations

Method	Parameter
FK_OperationRisk_Operation()	<u>VARCHAR2</u> name [in]
PK_OperationRisk()	<u>VARCHAR2</u> name [in]

3.5.4.4.5.5 OpEvent

Type: **Class**

Stereotype: «table»

Notes:

Notable events in the operation. Data here can be optional related to the OpAsset table to show the relationship between a service and an operational event.

Attributes

Name	Type	Length	Precision	Scale	Notes
opName	VARCHAR2	255	0	0	The operation name, linked to the operation table.
eventID	NUMBER	0	6	0	A sequence number identifier for the event.
created	TIMESTAMP	0	0	0	Timestamp when the record was created.
modified	TIMESTAMP	0	0	0	Timestamp when the record was last modified.
description	VARCHAR2	2000	0	0	General description of the event.
startTimestamp	TIMESTAMP	0	0	0	Start time of the event.
endTimestamp	TIMESTAMP	0	0	0	End time of the event.

Connections

Connector	Source	Target
<u>Association</u> (opName = opName eventID = eventID) Source -> Destination	Operation:OpDetail	Operation:OpEvent
<u>Association</u> (opName = name) Source -> Destination	Operation:OpEvent	Operation:Operation
<u>AssociationClass</u> Source -> Destination	Operation:OpEvent	C2IEDM:ACT_TASK

Operations

Method	Parameter
PK_OpEvent()	<u>VARCHAR2</u> opName [in] <u>NUMBER</u> eventID [in]
FK_OpEvent_Operation()	<u>VARCHAR2</u> opName [in]
TRG_PreInsert_OpEvent before insert()	
TRG_PreUpdate_opevent before update of opname()	

3.5.4.4.5.6 OpUnit

Type: **Class**

Stereotype: «table»

Notes:

Units involved in the operation are recorded. This is a simple representation that does not maintain a master reference of units.

Attributes

Name	Type	Length	Precision	Scale	Notes
opName	VARCHAR2	255	0	0	The operation name.
unitName	VARCHAR2	255	0	0	The unit name.
created	TIMESTAMP	0	0	0	Timestamp when the record was created.
modified	TIMESTAMP	0	0	0	Timestamp when the record was last modified.
unitID	NUMBER	0	15	0	Used to correlate with external data sources (C2IEDM).

Connections

Connector	Source	Target
<u>Association</u> (opName = name) Source -> Destination	Operation:OpUnit	Operation:Operation
<u>Association</u> (opName = opName unitName = unitName) Source -> Destination	Operation:OpDetail	Operation:OpUnit
<u>AssociationClass</u> Source -> Destination	Operation:OpUnit	C2IEDM:ORG

Operations

Method	Parameter
PK_Unit()	<u>VARCHAR2</u> opName [in] <u>VARCHAR2</u> unitName [in]
FK_Unit_Operation()	<u>VARCHAR2</u> opName [in]
TRG_PreInsert_OpUnit before insert()	
TRG_PreUpdate_opunit before update of opname()	

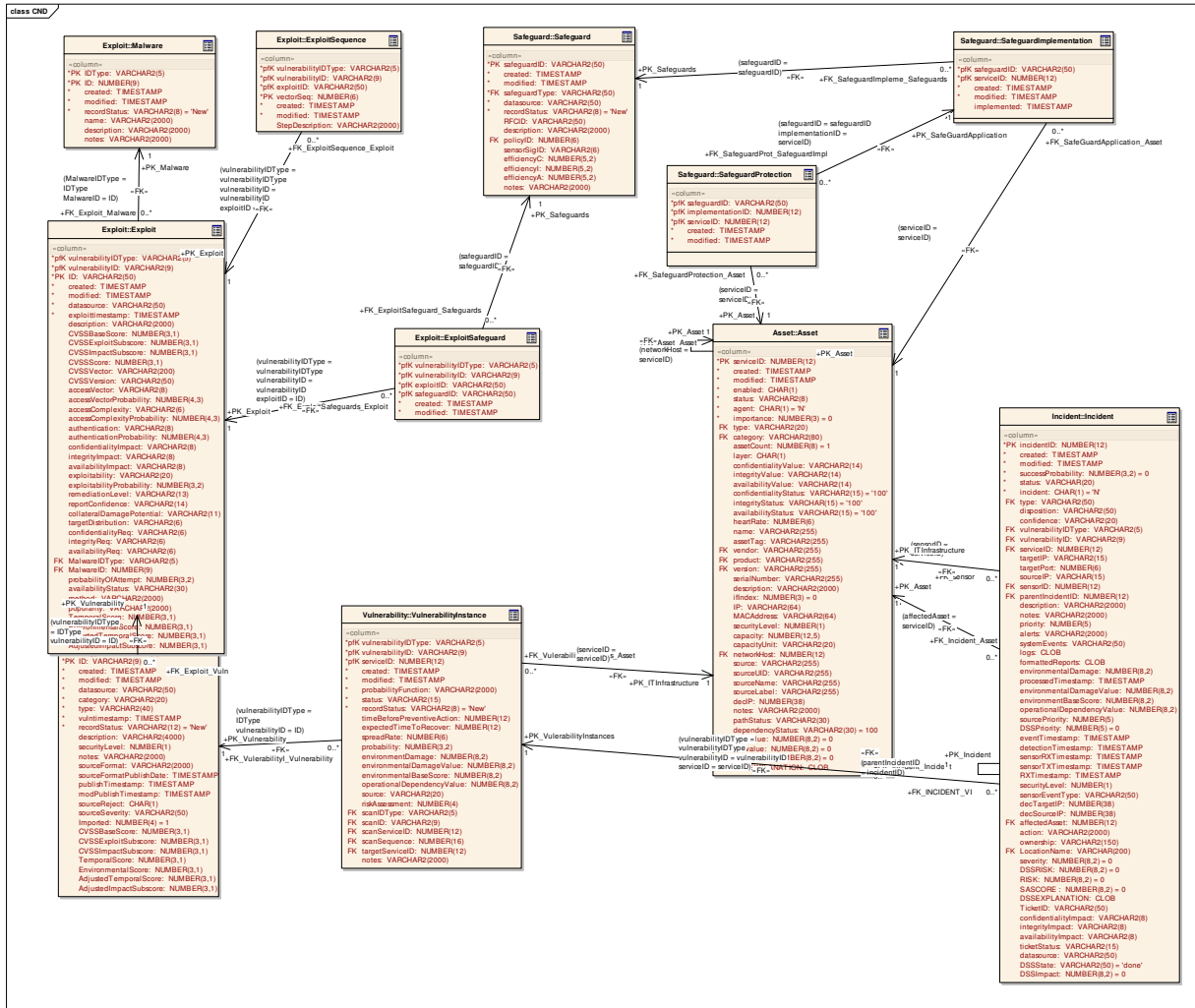
3.5.4.5 CND

The data security domain deals with modeling threats. Modeling threats identifies the data associated with a given threat, and when related to the IT infrastructure data provides a view to system administrators of the impact of a threat on the system.

Security Events Data will be captured into the database with these inputs:

1. Security Advisories
2. Alerts and Admin Notes
3. Network Forecasted Events
4. Intelligence Data
5. Network Devices, Logs and Alerts
6. Security Devices, Logs and Alerts
7. Shared Network Incident Information

CND - (Logical diagram)



CND High Level - (Logical diagram)

This diagram depicts malware and vulnerability definitions are 2 independent entities.

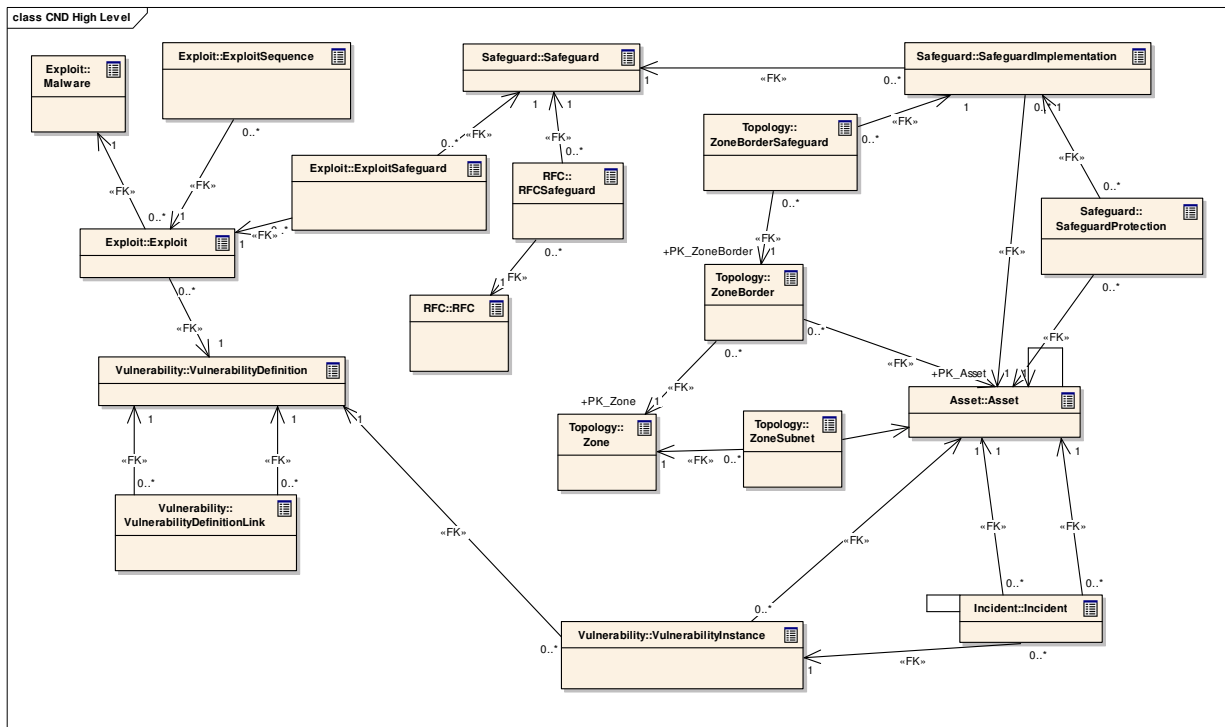
Exploits are children of vulnerabilities.

A vulnerability will have at least 1 exploit "unknown". (enforced in the app, not the DB)

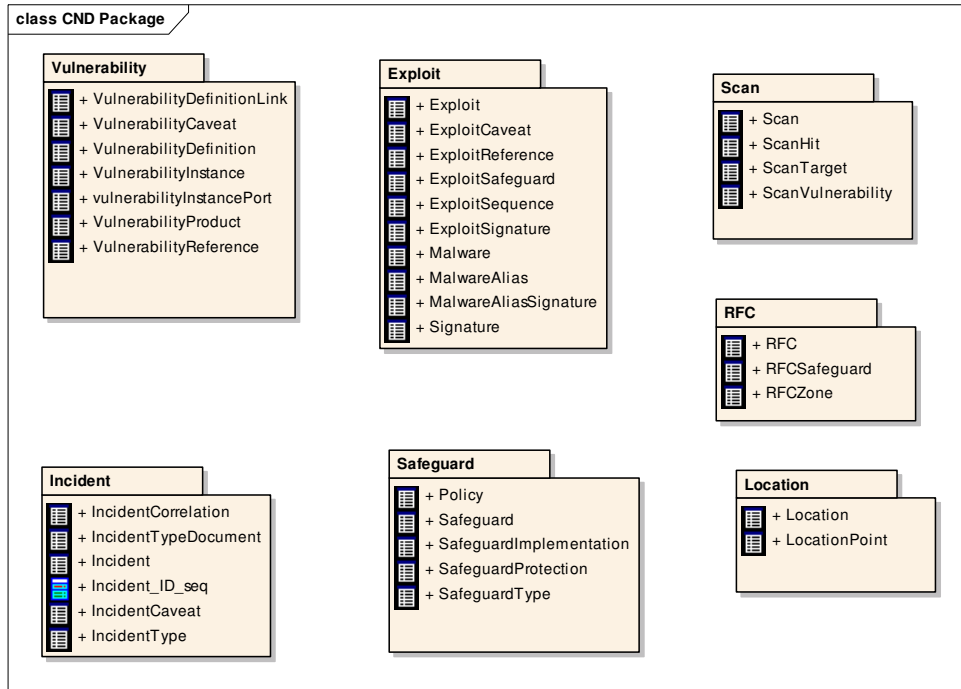
An exploit may be realized in 0 or more malwares.

A malware must implement 1 or more exploits. (This will be an implementation detail, for the purposes of the data model, we will not enforce the referential integrity required to force that at least 1 exploit must exist for a given malware instance.

An exploit is a method, not a software item. If a single software item exists for that exploit, then it will be contained in the malware table and linked accordingly.



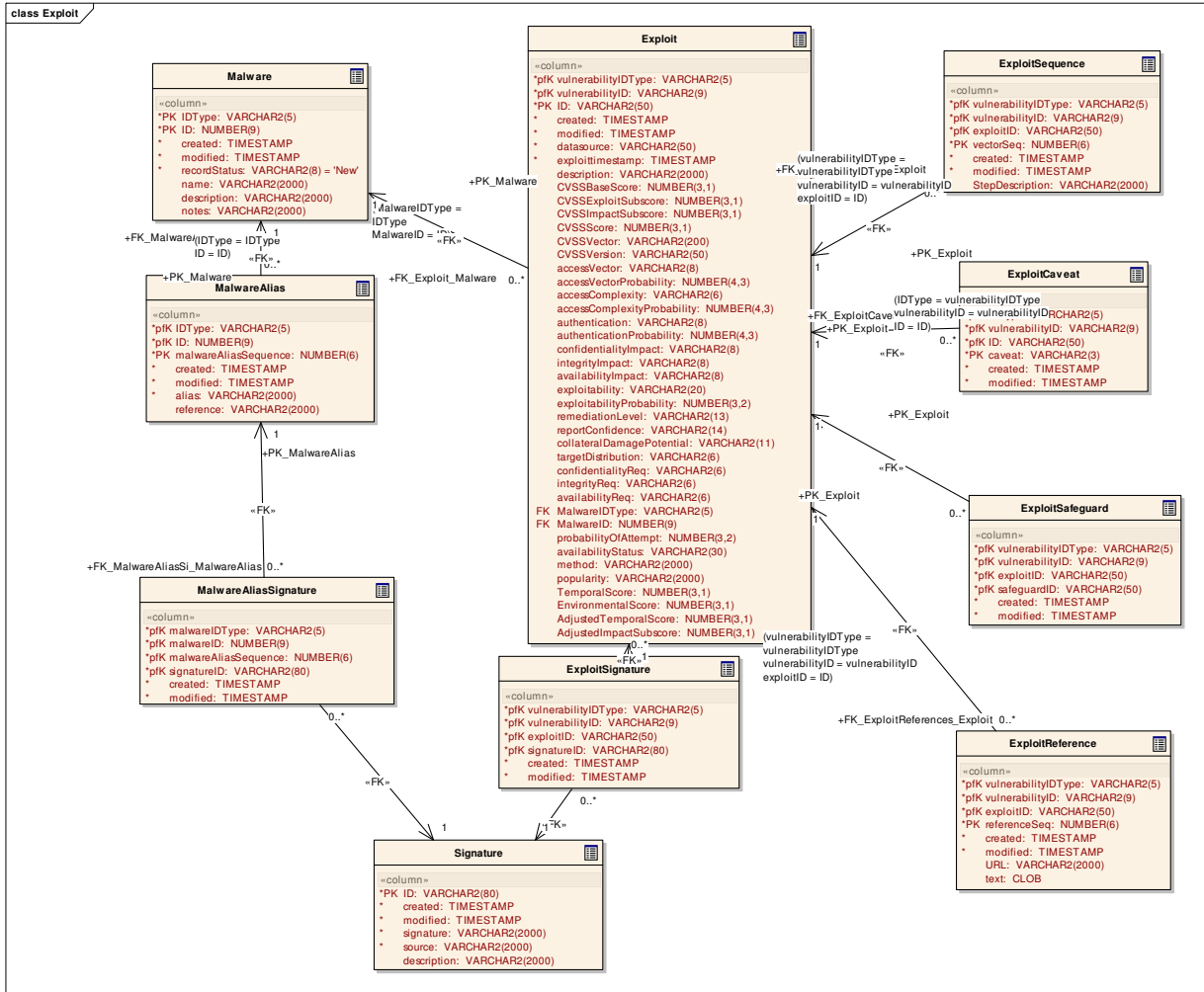
CND Package - (Logical diagram)



3.5.4.5.1 Exploit

The Exploit Package contains entities related to Exploits

Exploit - (Logical diagram)



3.5.4.5.1.1 Exploit

Type: **Class**

Stereotype: «table»

Notes:

The exploit contains information on any exploits associated with the vulnerability. The exploit is the "method", not the software. Software is defined in the Malware table and related via a relationship. Every vulnerability will have at least 1 exploit, the default being "unknown" which would be created by the software automatically (operator input not required).

Attributes

Name	Type	Length	Precision	Scale	Notes
vulnerabilityIDType	VARCHAR2	5	0	0	Vulnerability ID type.
vulnerabilityID	VARCHAR2	9	0	0	Vulnerability ID.
ID	VARCHAR2	50	0	0	Exploit ID.
created	TIMESTAMP	0	0	0	Timestamp record was created.
modified	TIMESTAMP	0	0	0	Timestamp of last modification.
datasource	VARCHAR2	50	0	0	Data source for the exploit.
exploittimestamp	TIMESTAMP	0	0	0	Timestamp from exploit source.
description	VARCHAR2	2000	0	0	Free text description of the exploit.
CVSSBaseScore	NUMBER		3	1	CVSS version 2 Base Score (nvd.entry.CVSS_base_score)
CVSSExploitSubscore	NUMBER		3	1	20* AccessVector*AccessComplexity* Authentication
CVSSImpactSubscore	NUMBER		3	1	10.41*(1-(1- confidentialityImpact)*(1- integrityImpact)*(1- availabilityImpact))

DN0678
Issue 4/2: 06 February 2012

Name	Type	Length	Precision	Scale	Notes
CVSSScore	NUMBER		3	1	CVSS Score reported on the exploit. (nvd.entry.CVSS_score)
CVSSVector	VARCHAR2	200			CVSS Access Vector string. (nvd.entry.CVSS_vector)
CVSSVersion	VARCHAR2	50			CVSS Version Indicator (nvd.entry.CVSS_version)
accessVector	VARCHAR2	8			Local: The vulnerability is only exploitable locally (i.e., it requires physical access or authenticated login to the target system) Network: The vulnerability is exploitable remotely Adjacent: Exploiting the vulnerability requires access to the local network of the target
accessVectorProbability	NUMBER		4	3	Probability value for the access vector.
accessComplexity	VARCHAR2	6			The complexity of attack required to exploit the vulnerability once an attacker has access to the target system. High Medium Low
accessComplexityProbability	NUMBER		4	3	Probability associated with the Access Complexity.

DN0678
Issue 4/2: 06 February 2012

Name	Type	Length	Precision	Scale	Notes
authentication	VARCHAR2	8			<p>Whether or not an attacker needs to be authenticated to the target system in order to exploit the vulnerability.</p> <p>Multiple: Two or more instances of authentication are required to exploit the vulnerability, even if the same credentials are used each time</p> <p>Single: One instance of authentication is required to exploit the vulnerability</p> <p>None: Authentication is not required to exploit the vulnerability</p>
authenticationProbability	NUMBER		4	3	Probability associated with the authentication.
confidentialityImpact	VARCHAR2	8	0	0	<p>The impact on confidentiality of a successful exploit of the vulnerability on the target system.</p> <p>None</p> <p>Partial</p> <p>Complete</p>
integrityImpact	VARCHAR2	8	0	0	<p>The impact on integrity a successful exploit of the vulnerability will have on the target system.</p> <p>None</p> <p>Partial</p> <p>Complete</p>
availabilityImpact	VARCHAR2	8	0	0	<p>The impact on availability a successful exploit of the vulnerability will have on the target system.</p> <p>None</p> <p>Partial</p> <p>Complete</p>

DN0678
Issue 4/2: 06 February 2012

Name	Type	Length	Precision	Scale	Notes
exploitability	VARCHAR2	20			<p>This metric attempts to measure the current state of exploit technique or code availability and suggests a likelihood of exploitation.</p> <p>Unproven</p> <p>Proof of Concept</p> <p>Functional</p> <p>Highly Functional</p>
exploitabilityProbability	NUMBER		3	2	<p>This probability will come from the DSS Load Values on initial load but can then be altered by an analyst for "special" cases.</p>
remediationLevel	VARCHAR2	13			<p>Measures the level of an available solution.</p> <p>Official Fix: Complete vendor solution available</p> <p>Temporary Fix: There is an official temporary fix available</p> <p>Workaround: There is an unofficial non-vendor solution available</p> <p>Unavailable: There is either no solution available or it is impossible to apply</p>
reportConfidence	VARCHAR2	14			<p>Measures the degree of confidence in the existence of the vulnerability and the credibility of its report.</p> <p>Unconfirmed: A single unconfirmed source or possibly multiple conflicting reports</p> <p>Uncorroborated: Multiple non-official sources; possibly including independent security companies or research organizations</p>

Name	Type	Length	Precision	Scale	Notes
					Confirmed: Vendor has reported/confirmed a problem with its own product, or an external event such as widespread exploitation confirms the existence of the problem
collateralDamage Potential	VARCHAR2	11			<p>Measures the potential for a loss of life or physical assets through damage or theft of property or equipment.</p> <p>None: There is no potential for loss of life, physical assets, productivity or revenue</p> <p>Low: A successful exploit of this vulnerability may result in slight physical or property damage, or slight loss of revenue or productivity</p> <p>Low-Medium: A successful exploit of this vulnerability may result in moderate physical or property damage, or moderate loss of revenue or productivity</p> <p>Medium-High: A successful exploit of this vulnerability may result in significant physical or property damage or loss, or significant loss of revenue or productivity</p> <p>High: A successful exploit of this vulnerability may result in catastrophic physical or property damage and loss, or catastrophic loss of revenue or productivity</p>
targetDistribution	VARCHAR2	6			<p>Measures the percentage of vulnerable systems.</p> <p>None: No target systems exist, or targets are so highly specialized that they only exist in a laboratory setting (effectively 0% of the environment is at risk)</p>

DN0678
Issue 4/2: 06 February 2012

Name	Type	Length	Precision	Scale	Notes
					<p>Low: Targets exist inside the environment, but on a small scale (between 1% - 25% of the total environment is at risk)</p> <p>Medium: Targets exist inside the environment, but on a medium scale (between 26% - 75% of the total environment is at risk)</p> <p>High: Targets exist inside the environment on a considerable scale (between 76% - 100% of the total environment is at risk)</p>
confidentialityReq	VARCHAR2	6			<p>Allows confidentiality to be customized depending on the criticality of the affected IT asset.</p> <p>Low: Loss is likely to have only a limited adverse effect on the organization or individuals associated with the organization</p> <p>Medium: Loss is likely to have a serious adverse effect on the organization or individuals associated with the organization</p> <p>High: Loss is likely to have a catastrophic adverse effect on the organization or individuals associated with the organization</p>
integrityReq	VARCHAR2	6			<p>Allows integrity to be customized depending on the criticality of the affected IT asset.</p> <p>Low: Loss is likely to have only a limited adverse effect on the organization or individuals associated with the organization</p> <p>Medium: Loss is likely to have a serious adverse effect on the organization or individuals associated with the organization</p>

DN0678
Issue 4/2: 06 February 2012

Name	Type	Length	Precision	Scale	Notes
					High: Loss is likely to have a catastrophic adverse effect on the organization or individuals associated with the organization
availabilityReq	VARCHAR2	6			Allows availability to be customized depending on the criticality of the affected IT asset. Low: Loss is likely to have only a limited adverse effect on the organization or individuals associated with the organization Medium: Loss is likely to have a serious adverse effect on the organization or individuals associated with the organization High: Loss is likely to have a catastrophic adverse effect on the organization or individuals associated with the organization
MalwareIDType	VARCHAR2	5	0	0	ID Type for associated malware
MalwareID	NUMBER	0	9	0	Malware ID.
probabilityOfAttempt	NUMBER	0	3	2	Probability of exploit attempt.
availabilityStatus	VARCHAR2	30			The availability of the exploit. This field currently mirrors the expectations of the exploitability field.
method	VARCHAR2	2000	0	0	Method of infection.
popularity	VARCHAR2	2000	0	0	Popularity of the exploit.
TemporalScore	NUMBER		3	1	An adjusted score taking into account the characteristics of a vulnerability that change over time but not among user environments: exploitability, remediation level, report confidence.

Name	Type	Length	Precision	Scale	Notes
EnvironmentalScore	NUMBER		3	1	An adjusted score taking into account the characteristics of a vulnerability that are relevant and unique to a particular user's environment: collateral damage potential, target distribution, security requirements.
AdjustedTemporalScore	NUMBER		3	1	TemporalScore recomputed with the BaseScore's Impact sub-equation replaced with the AdjustedImpact equation
AdjustedImpactSubscore	NUMBER		3	1	The impact subscore considering the security requirements

Connections

Connector	Source	Target
<u>Dependency</u> Source -> Destination	Data Warehousing:Store Vulnerability and Exploit Data	Exploit:Exploit
<u>Association</u> (vulnerabilityIDType = vulnerabilityIDType vulnerabilityID = vulnerabilityID exploitID = ID) Source -> Destination	Exploit:ExploitReference	Exploit:Exploit
<u>Association</u> (IDType = vulnerabilityIDType vulnerabilityID = vulnerabilityID ID = ID) Source -> Destination	Exploit:ExploitCaveat	Exploit:Exploit
<u>Association</u> (vulnerabilityIDType = vulnerabilityIDType	Exploit:ExploitSafeguard	Exploit:Exploit

Connector	Source	Target
vulnerabilityID = vulnerabilityID exploitID = ID) Source -> Destination		
<u>Dependency</u> Source -> Destination	Data Transformation:Pre- Process Vulnerability and Exploit Data	Exploit:Exploit
<u>Association</u> (vulnerabilityIDType = vulnerabilityIDType vulnerabilityID = vulnerabilityID exploitID = ID) Source -> Destination	Exploit:ExploitSequence	Exploit:Exploit
<u>Association</u> (vulnerabilityIDType = vulnerabilityIDType vulnerabilityID = vulnerabilityID exploitID = ID) Source -> Destination	Exploit:ExploitSignature	Exploit:Exploit
<u>Association</u> (MalwareIDType = IDType MalwareID = ID) Source -> Destination	Exploit:Exploit	Exploit:Malware
<u>Association</u> (vulnerabilityIDType = IDType vulnerabilityID = ID) Source -> Destination	Exploit:Exploit	Vulnerability:VulnerabilityDef inition

Operations

Method	Parameter
FK_Exploit_Vuln()	<u>VARCHAR2</u> vulnerabilityIDType [in] <u>VARCHAR2</u> vulnerabilityID [in]
FK_Exploit_Malware()	<u>VARCHAR2</u> MalwareIDType [in] <u>NUMBER</u> MalwareID [in]
PK_Exploit()	<u>VARCHAR2</u> vulnerabilityIDType [in] <u>VARCHAR2</u> vulnerabilityID [in] <u>VARCHAR2</u> ID [in]
TRG_PreInsert_Exploit before insert()	

3.5.4.5.1.2 ExploitCaveat

Type: **Class**

Stereotype: «table»

Notes:

The country codes (caveats) for the exploit.

Attributes

Name	Type	Length	Precision	Scale	Notes
IDType	VARCHAR2	5			Vulnerability ID Type.
vulnerabilityID	VARCHAR2	9	0	0	Vulnerability ID.
ID	VARCHAR2	50	0	0	Exploit ID
caveat	VARCHAR2	3	0	0	Country code (caveat).
created	TIMESTAMP	0	0	0	Timestamp record was created.
modified	TIMESTAMP	0	0	0	Timestamp of last modification

Connections

Connector	Source	Target
Association (IDType = vulnerabilityIDType vulnerabilityID = vulnerabilityID ID = ID) Source -> Destination	Exploit:ExploitCaveat	Exploit:Exploit

Operations

Method	Parameter
FK_ExploitCaveats_Exploit()	<u>VARCHAR2</u> IDType [in] <u>VARCHAR2</u> vulnerabilityID [in] <u>VARCHAR2</u> ID [in]
PK_ExploitCaveats()	<u>VARCHAR2</u> IDType [in] <u>VARCHAR2</u> vulnerabilityID [in] <u>VARCHAR2</u> ID [in] <u>VARCHAR2</u> caveat [in]
TRG_PreInsert_ExploitCaveat before insert()	

3.5.4.5.1.3 ExploitReference

Type: **Class**

Stereotype: «table»

Notes:

List of reference data for an Exploit.

Attributes

Name	Type	Length	Precision	Scale	Notes
vulnerabilityIDType	VARCHAR2	5			Vulnerability ID type.
vulnerabilityID	VARCHAR2	9	0	0	Vulnerability ID.
exploitID	VARCHAR2	50	0	0	Exploit ID.
referenceSeq	NUMBER	0	6	0	Reference sequence number.
created	TIMESTAMP	0	0	0	Timestamp of record creation.
modified	TIMESTAMP	0	0	0	timestamp of last modification.
URL	VARCHAR2	2000	0	0	URL of reference.
text	CLOB	0	0	0	Free text of reference.

Connections

Connector	Source	Target
<u>Association</u> (vulnerabilityIDType = vulnerabilityIDType vulnerabilityID = vulnerabilityID exploitID = ID) Source -> Destination	Exploit:ExploitReference	Exploit:Exploit

Operations

Method	Parameter
FK_ExploitReferences_Exploit()	<u>VARCHAR2</u> vulnerabilityIDType [in] <u>VARCHAR2</u> vulnerabilityID [in] <u>VARCHAR2</u> exploitID [in]
PK_ExploitReferences()	<u>VARCHAR2</u> vulnerabilityIDType [in] <u>VARCHAR2</u> vulnerabilityID [in] <u>VARCHAR2</u> exploitID [in] <u>NUMBER</u> referenceSeq [in]
TRG_PreInsert_ExploitReference before insert()	

3.5.4.5.1.4 ExploitSafeguard

Type: **Class**

Stereotype: «table»

Notes:

Conceivably, a given exploit (exploit sequence) could have one or more safeguards applied. As an example, perhaps a portion of the network assets use McAfee and the rest is using Symantec (two different safeguards that would protect against the exploit). In the other direction, a firewall rule (such as blocking a specific port) would safeguard all exploits for that port.

Attributes

Name	Type	Length	Precision	Scale	Notes
vulnerabilityIDType	VARCHAR2	5			ID Type from the vulnerability Definition (i.e.. 'CVE').
vulnerabilityID	VARCHAR2	9	0	0	In ID of the vulnerability definition.
exploitID	VARCHAR2	50	0	0	The ID from the exploit.
safeguardID	VARCHAR2	50	0	0	The ID of the safeguard.
created	TIMESTAMP	0	0	0	Date record was created.
modified	TIMESTAMP	0	0	0	Date of last modification.

Connections

Connector	Source	Target
<u>Association</u> (safeguardID = safeguardID) Source -> Destination	Exploit:ExploitSafeguard	Safeguard:Safeguard
<u>Association</u> (vulnerabilityIDType = vulnerabilityIDType vulnerabilityID = vulnerabilityID exploitID = ID) Source -> Destination	Exploit:ExploitSafeguard	Exploit:Exploit

Operations

Method	Parameter
FK_ExploitSafeguards_Exploit()	<u>VARCHAR2</u> vulnerabilityIDType [in] <u>VARCHAR2</u> vulnerabilityID [in] <u>VARCHAR2</u> exploitID [in]
FK_ExploitSafeguard_Safeguards()	<u>VARCHAR2</u> safeguardID [in]
PK_ExploitSafeguards()	<u>VARCHAR2</u> vulnerabilityIDType [in] <u>VARCHAR2</u> vulnerabilityID [in] <u>VARCHAR2</u> exploitID [in] <u>VARCHAR2</u> safeguardID [in]
TRG_PreInsert_ExploitSafeguard before insert()	

3.5.4.5.1.5 ExploitSequence

Type: **Class**

Stereotype: «table»

Notes:

This entity allows the sequence/steps to be modeled. In this way, specific steps can be safeguarded and thus giving a start at modeling the attack vector. For the purposes of JNDMS TDP, this detail would probably not be presented to the user. We'll implement a hidden 1-to-1 relationship to the exploit. This table allows us to expand in this direction in the future.

Attributes

Name	Type	Length	Precision	Scale	Notes
vulnerabilityIDType	VARCHAR2	5			Foreign key to the Exploit table
vulnerabilityID	VARCHAR2	9			Foreign key to the Exploit table
exploitID	VARCHAR2	50			Foreign key to the Exploit table
vectorSeq	NUMBER		6	0	Primary key for identifying the sequence of the exploitation steps
created	TIMESTAMP	0	0	0	Date record was created.
modified	TIMESTAMP	0	0	0	Date of last modification
StepDescription	VARCHAR2	2000	0	0	A text based description of the step.

Connections

Connector	Source	Target
<u>Association</u> (vulnerabilityIDType = vulnerabilityIDType vulnerabilityID = vulnerabilityID exploitID = ID) Source -> Destination	Exploit:ExploitSequence	Exploit:Exploit

Operations

Method	Parameter
FK_ExploitSequence_Exploit()	<u>VARCHAR2</u> vulnerabilityIDType [in] <u>VARCHAR2</u> vulnerabilityID [in] <u>VARCHAR2</u> exploitID [in]
PK_attackVector()	<u>VARCHAR2</u> vulnerabilityIDType [in] <u>VARCHAR2</u> vulnerabilityID [in] <u>VARCHAR2</u> exploitID [in] <u>NUMBER</u> vectorSeq [in]
TRG_PreInsert_ExploitSequence before insert()	

3.5.4.5.1.6 ExploitSignature

Type: **Class**

Stereotype: «table»

Notes:

This table stores signatures that may be related directly to an exploit.

Attributes

Name	Type	Length	Precision	Scale	Notes
vulnerabilityIDType	VARCHAR2	5			ID Type from the vulnerability Definition (i.e.. 'CVE').
vulnerabilityID	VARCHAR2	9	0	0	In ID of the vulnerability definition.
exploitID	VARCHAR2	50	0	0	The ID from the exploit.
signatureID	VARCHAR2	80	0	0	ID for the signature.
created	TIMESTAMP	0	0	0	Timestamp of record creation.
modified	TIMESTAMP	0	0	0	Timestamp of last modification.

Connections

Connector	Source	Target
<u>Association</u> (vulnerabilityIDType = vulnerabilityIDType vulnerabilityID = vulnerabilityID exploitID = ID) Source -> Destination	Exploit:ExploitSignature	Exploit:Exploit
<u>Association</u> (signatureID = ID) Source -> Destination	Exploit:ExploitSignature	Exploit:Signature

Operations

Method	Parameter
FK_ExploitSignature_Exploit()	<u>VARCHAR2</u> vulnerabilityIDType [in] <u>VARCHAR2</u> vulnerabilityID [in] <u>VARCHAR2</u> exploitID [in]
FK_ExploitSignature_signature()	<u>VARCHAR2</u> signatureID [in]
PK_ExploitSignature()	<u>VARCHAR2</u> vulnerabilityIDType [in] <u>VARCHAR2</u> vulnerabilityID [in] <u>VARCHAR2</u> exploitID [in] <u>VARCHAR2</u> signatureID [in]
TRG_PreIns_ExploitSig before insert()	

3.5.4.5.2 Malware

Type: Class

Stereotype: «table»

Notes:

The Malware is the "software". It would be filled by the CME list + other sources. It may implement one or more exploits. An exploit may be implemented in one or more malware.

Attributes

Name	Type	Length	Precision	Scale	Notes
IDType	VARCHAR2	5			Hold the acronym portion of the identifier, "CME".
ID	NUMBER		9	0	The identifier as described on http://cme.mitre.org/cme/process.html shows that there can be up to 7 digits but that leading zeroes will not be displayed.
created	TIMESTAMP	0	0	0	Date record was created.
modified	TIMESTAMP	0	0	0	Date of last modification
recordStatus	VARCHAR2	8			<p>The record status is used to indicate JNDMS internal handling of the data entity. It can be used to indicate to the operators that action is required and may be used by the system to determine whether or not the data is being used in processing. Allowable values include:</p> <p>New - New record that should be confirmed by operators.</p> <p>Pending - A record that is currently under review/being researched.</p> <p>Approved - A record that is approved for use in the JNDMS. Approval may come manually or via automated processing depending upon the data/scenario.</p> <p>Denied - Data that is retained but not used by the JNDMS system.</p>

DN0678
Issue 4/2: 06 February 2012

Name	Type	Length	Precision	Scale	Notes
name	VARCHAR2	2000	0	0	The properties that identify the "malware" and can be used for detection/protection.
description	VARCHAR2	2000	0	0	Description of the Malware
notes	VARCHAR2	2000			Notes entered by the operator or by automated processing about the record.

Connections

Connector	Source	Target
<u>Association</u> (IDType = IDType ID = ID) Source -> Destination	Exploit:MalwareAlias	Exploit:Malware
<u>Association</u> (MalwareIDType = IDType MalwareID = ID) Source -> Destination	Exploit:Exploit	Exploit:Malware

Operations

Method	Parameter
PK_Malware()	<u>VARCHAR2</u> IDType [in] <u>NUMBER</u> ID [in]
TRG_PreInsert_Malware before insert()	

3.5.4.5.2.1 MalwareAlias

Type: **Class**

Stereotype: «table»

Notes:

The list aliases for a given malware.

Attributes

Name	Type	Length	Precision	Scale	Notes
IDType	VARCHAR2	5	0	0	ID Type from the parent malware.
ID	NUMBER	0	9	0	ID from the parent malware.
malwareAliasSequence	NUMBER	0	6	0	Sequence within the parent.
created	TIMESTAMP	0	0	0	Timestamp record was created.
modified	TIMESTAMP	0	0	0	Timestamp of last modification.
alias	VARCHAR2	2000	0	0	The alias name.
reference	VARCHAR2	2000	0	0	Reference and or link.

Connections

Connector	Source	Target
Association (IDType = IDType ID = ID) Source -> Destination	Exploit:MalwareAlias	Exploit:Malware
Association (malwareIDType = IDType malwareID = ID malwareAliasSequence = malwareAliasSequence) Source -> Destination	Exploit:MalwareAliasSignature	Exploit:MalwareAlias

Operations

Method	Parameter
FK_MalwareAlias_Malware()	<u>VARCHAR2</u> IDType [in] <u>NUMBER</u> ID [in]
PK_MalwareAlias()	<u>VARCHAR2</u> IDType [in] <u>NUMBER</u> ID [in] <u>NUMBER</u> malwareAliasSequence [in]
TRG_PreInsert_MalwareAlias before insert()	

3.5.4.5.2.2 MalwareAliasSignature

Type: **Class**

Stereotype: «table»

Notes:

Attributes

Name	Type	Length	Precision	Scale	Notes
malwareIDType	VARCHAR2	5	0	0	ID Type from the parent malware.
malwareID	NUMBER	0	9	0	ID from the parent malware.
malwareAliasSequence	NUMBER	0	6	0	Sequence within the parent.
signatureID	VARCHAR2	80	0	0	The ID from the exploit.
created	TIMESTAMP	0	0	0	Timestamp of record creation.
modified	TIMESTAMP	0	0	0	Timestamp of last modification.

Connections

Connector	Source	Target
Association (malwareIDType = IDType malwareID = ID malwareAliasSequence = malwareAliasSequence) Source -> Destination	Exploit:MalwareAliasSignature	Exploit:MalwareAlias
Association (signatureID = ID) Source -> Destination	Exploit:MalwareAliasSignature	Exploit:Signature

Operations

Method	Parameter
FK_MalwareAliasSi_MalwareAlias()	<u>VARCHAR2</u> malwareIDType [in] <u>NUMBER</u> malwareID [in] <u>NUMBER</u> malwareAliasSequence [in]
FK_MalwareAliasSigna_signature()	<u>VARCHAR2</u> signatureID [in]
PK_MalwareAliasSignature()	<u>VARCHAR2</u> malwareIDType [in] <u>NUMBER</u> malwareID [in] <u>NUMBER</u> malwareAliasSequence [in] <u>VARCHAR2</u> signatureID [in]
TRG_PreIns_MalAliasSig before insert()	

3.5.4.5.3 Signature

Type: Class

Stereotype: «table»

Notes:

The signature table contains a mixture of signatures. Some signatures represent malware or "bad things" that can be detected by a virus scanner, email scanners, and other detection tools. Other signatures represent exploits or "bad actions" which are typically only detected by the IDS or firewall. An example is the "Ping of Death" which is not a malware but can be detected "on the wire" as a malformed ICMP packet.

Attributes

Name	Type	Length	Precision	Scale	Notes
ID	VARCHAR2	80	0	0	The ID can be "home grown" or provided by the "sensor" or even be the vulnerability ID (CVE-9999-1234)
created	TIMESTAMP	0	0	0	Timestamp of record creation.
modified	TIMESTAMP	0	0	0	Timestamp of last modification.
signature	VARCHAR2	2000	0	0	The text to match the signature on.
source	VARCHAR2	2000	0	0	Signature source
description	VARCHAR2	2000	0	0	Textual description of the signature.

Connections

Connector	Source	Target
<u>Association</u> (signatureID = ID) Source -> Destination	Exploit:ExploitSignature	Exploit:Signature
<u>Association</u> (signatureID = ID) Source -> Destination	Exploit:MalwareAliasSignature	Exploit:Signature

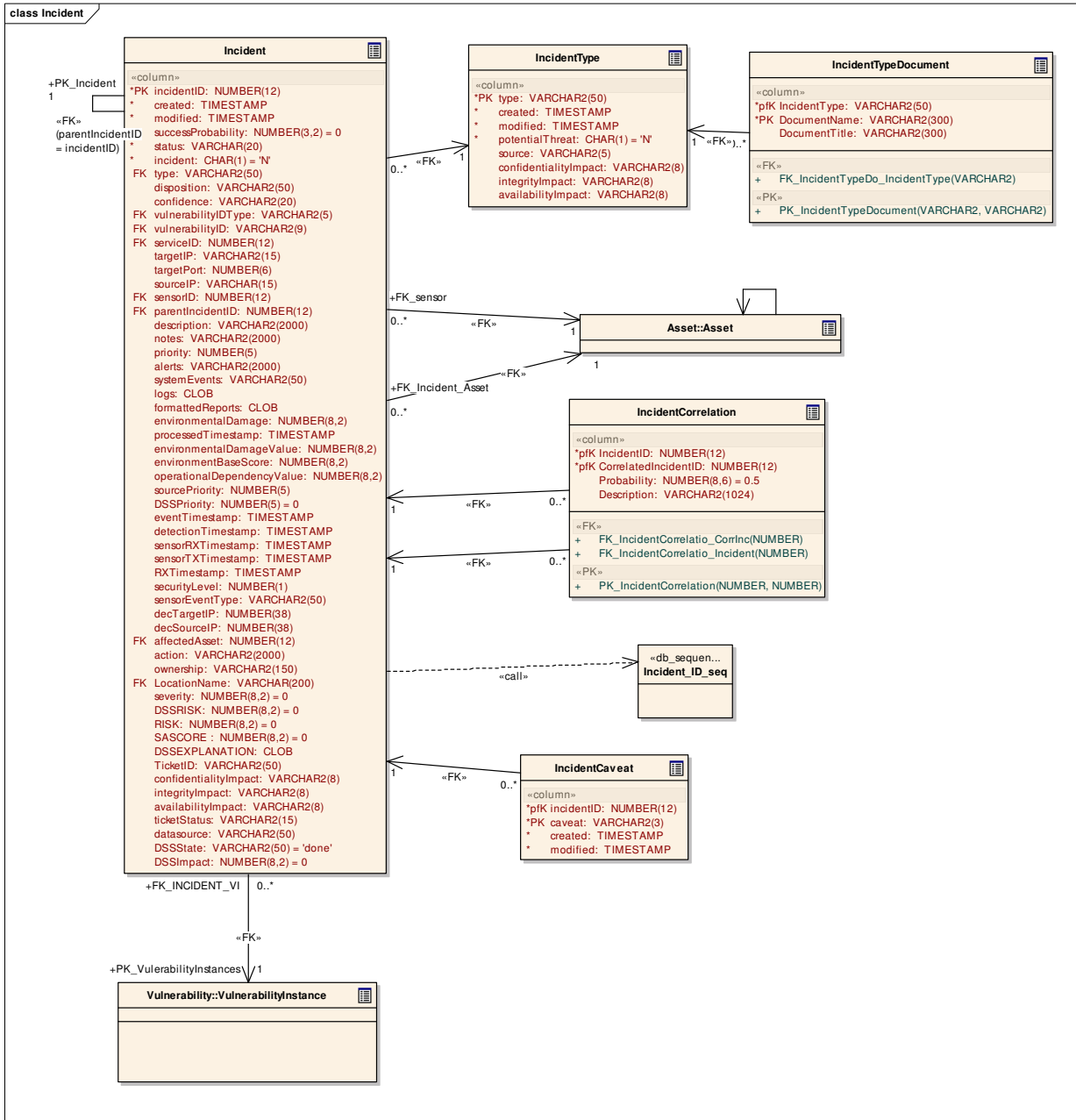
Operations

Method	Parameter
PK_signature()	<u>VARCHAR2</u> ID [in]
TRG_PreInsert_Signature before insert()	

3.5.4.5.4 Incident

The Incident Package provides details related to the incident entity.

Incident - (Logical diagram)



3.5.4.5.4.1 Incident

Type: **Class**

Stereotype: «table»

Notes:

The incident table records the incidents that are passed to JNDMS via its sensor or as recorded by the operators.

Attributes

Name	Type	Length	Precision	Scale	Notes
incidentID	NUMBER	0	12	0	Sequence number with the vulnerability instance.
created	TIMESTAMP	0	0	0	Timestamp record was created.
modified	TIMESTAMP	0	0	0	Timestamp of last modification.
successProbability	NUMBER	0	3	2	The probability that the incident will result in a successful exploitation of the system.
status	VARCHAR	20	0	0	Incident status. Active Mitigated Resolved Forecasted
incident	CHAR	1	0	0	'Y' or 'N'. 'Y' indicates that it is classified by the DSS (or operator) as an incident (has CIA impact) or as an event.
type	VARCHAR2	50	0	0	Allowable values include "Security" and "Infrastructure". Not sure how Remedy/Trouble Tickets fit in the mix here.
disposition	VARCHAR2	50	0	0	Set from a list of allowable values, indicates what the incident is classified as (threat, violation, safeguarded,...)

DN0678
Issue 4/2: 06 February 2012

Name	Type	Length	Precision	Scale	Notes
confidence	VARCHAR2	20	0	0	This will hold the level of assessment completed. For example, initial status could be "Detection" and then it gets set to "Level 1" when the level 1 CIRT analyst has completed his task. Users could use this Confidence as a filter on what data they wish to see. For example, the Watch Officer may only want is display to consider incidents that have completed "Level 2".
vulnerabilityIDType	VARCHAR2	5			Vulnerability ID type.
vulnerabilityID	VARCHAR2	9	0	0	Vulnerability ID.
serviceID	NUMBER	0	12	0	Service ID of the affected asset. This attribute is overloaded. It will be filled directly with the asset service ID upon initial load of the incident. Upon correlation with a vulnerability instance, the data will be related via that foreign key.
targetIP	VARCHAR2	15	0	0	999.999.999.999
targetPort	NUMBER	0	6	0	Targeted port number.
sourceIP	VARCHAR	15	0	0	Format mask 999.999.999.999
sensorID	NUMBER	0	12	0	Service ID of the asset that sensed the incident.
parentIncidentID	NUMBER	0	12	0	Used to maintain a hierarchy of incidents.
description	VARCHAR2	2000	0	0	Free text description of the incident.
notes	VARCHAR2	2000	0	0	Analysts notes.
priority	NUMBER		5	0	Simple capture of the priority of the incident as generated by the DSS and/or by operator override.
alerts	VARCHAR2	2000	0	0	Free text list of alerts.

DN0678
Issue 4/2: 06 February 2012

Name	Type	Length	Precision	Scale	Notes
systemEvents	VARCHAR2	50	0	0	System events reported from the source.
logs	CLOB	0	0	0	Any log files associated with the incident.
formattedReports	CLOB	0	0	0	Any formatted reports on the incident.
environmentalDamage	NUMBER	0	8	2	Environmental damage score for the incident.
processedTimestamp	TIMESTAMP	0	0	0	Timestamp recording completion of DSS processing of the incident.
environmentalDamageValue	NUMBER	0	8	2	Value of the environmental damage.
environmentBaseScore	NUMBER	0	8	2	Base value for the environment.
operationalDependencyValue	NUMBER	0	8	2	Score of the operational dependency of the incident.
sourcePriority	NUMBER		5	0	Priority assigned by the incident source.
DSSPriority	NUMBER		5	0	DSS raw severity/impact score
eventTimestamp	TIMESTAMP	0	0	0	This is the timestamp of the incident as recorded by the sensor.
detectionTimestamp	TIMESTAMP	0	0	0	Timestamp the initial sensor made the detection.
sensorRXTimestamp	TIMESTAMP	0	0	0	Timestamp that the incident was received at the JNDMS sensor level.
sensorTXTimestamp	TIMESTAMP	0	0	0	Timestamp when the JNDMS sensor issued the incident to JNDMS.
RXTimestamp	TIMESTAMP	0	0	0	Timestamp when JNDMS received the incident.
securityLevel	NUMBER	0	1	0	Security level of the incident.

DN0678
Issue 4/2: 06 February 2012

Name	Type	Length	Precision	Scale	Notes
sensorEventType	VARCHAR2	50	0	0	The event type as reported by the sensor.
decTargetIP	NUMBER	0	38	0	Decimal equivalent to the target IP.
decSourceIP	NUMBER	0	38	0	Decimal equivalent to the source IP
affectedAsset	NUMBER	0	12	0	An asset can be linked to the incident without a vulnerability. Note, that if both are not null, they must be equal.
action	VARCHAR2	2000			Written description of the action taken.
ownership	VARCHAR2	150			
LocationName	VARCHAR	200			The name of the location of the incident.
severity	NUMBER		8	2	severity/impact score normalized to a 0-100 range
DSSRISK	NUMBER		8	2	default DSS raw risk score
RISK	NUMBER		8	2	default risk score normalized to a 0-100 range
SASCORE	NUMBER		8	2	“Situational Awareness” score, combines Risk and Impact
DSSEXPLANATION	CLOB				Text explaining DSS "evidence" to support severity/risk "score"
TicketID	VARCHAR2	50			An ID for an external ticketing system.
confidentialityImpact	VARCHAR2	8			Copied from the Incident type so the user can customize it case-by-case
integrityImpact	VARCHAR2	8			Copied from the incidentType table so the user can customize
availabilityImpact	VARCHAR2	8			Copied from the incidentType table so the user can customize
ticketStatus	VARCHAR2	15			The status of the external ticket

Name	Type	Length	Precision	Scale	Notes
datasource	VARCHAR2	50			Data source of the event information
DSSState	VARCHAR2	50			Allow for the DSS to update the state of processing risk. Values allowed include 'pending', 'processing', and 'done'
DSSImpact	NUMBER		8	2	Normalized Impact value based on a combination of OpAsset Impacts

Connections

Connector	Source	Target
Association (type = type) Source -> Destination	Incident:Incident	Incident:IncidentType
Dependency Source -> Destination	Incident:Incident	Incident:Incident_ID_seq
Association (parentIncidentID = incidentID) Unspecified	Incident:Incident	Incident:Incident
Dependency Source -> Destination	Security Information Management:Correlate Security Events and Vulnerability Scans	Incident:Incident
Association (incidentID = incidentID) Source -> Destination	Incident:IncidentCaveat	Incident:Incident
Dependency Source -> Destination	Security Information Management:Detection of traffic pattern raises alarms	Incident:Incident
Association (vulnerabilityIDType = vulnerabilityIDType vulnerabilityID = vulnerabilityID serviceID = serviceID) Source -> Destination	Incident:Incident	Vulnerability:VulnerabilityInstance

Connector	Source	Target
<u>Association</u> (LocationName = LocationName) Source -> Destination	Incident:Incident	Location:Location
<u>Association</u> (CorrelatedIncidentID = incidentID) Source -> Destination	Incident:IncidentCorrelation	Incident:Incident
<u>Association</u> (IncidentID = incidentID) Source -> Destination	Incident:IncidentCorrelation	Incident:Incident
<u>Association</u> (affectedAsset = serviceID) Source -> Destination	Incident:Incident	Asset:Asset
<u>Association</u> (sensorID = serviceID) Source -> Destination	Incident:Incident	Asset:Asset

Operations

Method	Parameter
FK_Incident_IncidentType()	<u>VARCHAR2</u> type [in]
FK_Incident_Location()	<u>VARCHAR</u> LocationName [in]
FK_Incident_Asset()	<u>NUMBER</u> affectedAsset [in]
FK_Incident_Incident()	<u>NUMBER</u> parentIncidentID [in]
FK_INCIDENT_VI()	<u>VARCHAR2</u> vulnerabilityIDType [in] <u>VARCHAR2</u> vulnerabilityID [in] <u>NUMBER</u> serviceID [in]
FK_sensor()	<u>NUMBER</u> sensorID [in]
PK_Incident()	<u>NUMBER</u> incidentID [in]
TRG_PreInsert_Incident before insert()	
CHK_affAsset_VI()	
TRG_Incident_ChangeLog before insert or update()	
TRG_IPUupdate_Incident before insert or update()	

3.5.4.5.4.2 Incident_ID_seq

Type: **Class**

Stereotype: «db_sequence»

Notes:

Connections

Connector	Source	Target
<u>Dependency</u> Source -> Destination	Incident:Incident	Incident:Incident_ID_seq

3.5.4.5.4.3 IncidentCaveat

Type: **Class**

Stereotype: «table»

Notes:

The list of countries (caveats) for the incident.

Attributes

Name	Type	Length	Precision	Scale	Notes
incidentID	NUMBER	0	12	0	The incident ID.
caveat	VARCHAR2	3	0	0	Country code (caveat).
created	TIMESTAMP	0	0	0	Timestamp of record creation.
modified	TIMESTAMP	0	0	0	Timestamp of last modification.

Connections

Connector	Source	Target
<u>Association</u> (incidentID = incidentID) Source -> Destination	Incident:IncidentCaveat	Incident:Incident

Operations

Method	Parameter
FK_IncidentCaveats_Incident()	<u>NUMBER</u> incidentID [in]
PK_IncidentCaveats()	<u>NUMBER</u> incidentID [in] <u>VARCHAR2</u> caveat [in]
TRG_PreInsert_IncidentCaveat before insert()	

3.5.4.5.4.4 IncidentCorrelation

Type: **Class**

Stereotype: «table»

Notes:

This table stores indirect correlations between events. These are fuzzy correlations inferred by the system (which does not mean there is a definite cause & effect relationship between the two events).

Attributes

Name	Type	Length	Precision	Scale	Notes
IncidentID	NUMBER		12	0	The parent incident ID for the correlation
CorrelatedIncidentID	NUMBER		12	0	The correlated event
Probability	NUMBER		8	6	The percentage of how corellated they are
Description	VARCHAR2	1024			Store a description of the match. The extra 24 chars are for Bill

Connections

Connector	Source	Target
<u>Association</u> (CorrelatedIncidentID = incidentID) Source -> Destination	Incident:IncidentCorrelation	Incident:Incident
<u>Association</u> (IncidentID = incidentID) Source -> Destination	Incident:IncidentCorrelation	Incident:Incident

Operations

Method	Parameter
FK_IncidentCorrelatio_CorrInc()	<u>NUMBER</u> CorrelatedIncidentID [in]
FK_IncidentCorrelatio_Incident()	<u>NUMBER</u> IncidentID [in]
PK_IncidentCorrelation()	<u>NUMBER</u> IncidentID [in] <u>NUMBER</u> CorrelatedIncidentID [in]

3.5.4.5.4.5 IncidentType

Type: **Class**

Stereotype: «table»

Notes:

The incident types are modeled as a look-up.

Attributes

Name	Type	Length	Precision	Scale	Notes
type	VARCHAR2	50	0	0	Incident Type
created	TIMESTAMP	0	0	0	Timestamp when record was created.
modified	TIMESTAMP	0	0	0	Timestamp of last modification.
potentialThreat	CHAR	1			The incident indicates a potential threat. Y or N.
source	VARCHAR2	5	0	0	Incident source type. TT EIM SIM DSS JNDMS
confidentialityImpact	VARCHAR2	8	0	0	Impact on confidentiality. None Partial Complete
integrityImpact	VARCHAR2	8	0	0	Impact on integrity. None Partial Complete

Name	Type	Length	Precision	Scale	Notes
availabilityImpact	VARCHAR2	8	0	0	Availability impact. None Partial Complete

Connections

Connector	Source	Target
<u>Association</u> (type = type) Source -> Destination	Incident:Incident	Incident:IncidentType
<u>Association</u> (IncidentType = type) Source -> Destination	Incident:IncidentTypeDocument	Incident:IncidentType

Operations

Method	Parameter
PK_IncidentType()	<u>VARCHAR2</u> type [in]
TRG_PreInsert_IncidentType before insert()	

3.5.4.5.4.6 IncidentTypeDocument

Type: **Class**

Stereotype: «table»

Notes:

Link between Jackrabbit DocumentLibrary documents (external) and the incidents.

Attributes

Name	Type	Length	Precision	Scale	Notes
IncidentType	VARCHAR2	50			Foreign key to IncidentType table
DocumentName	VARCHAR2	300			The 'Name' of the JCR document. This doesn't match the displayed name, but takes the form of 'DLFE-1.htm'
DocumentTitle	VARCHAR2	300			The name of the file - user readable

Connections

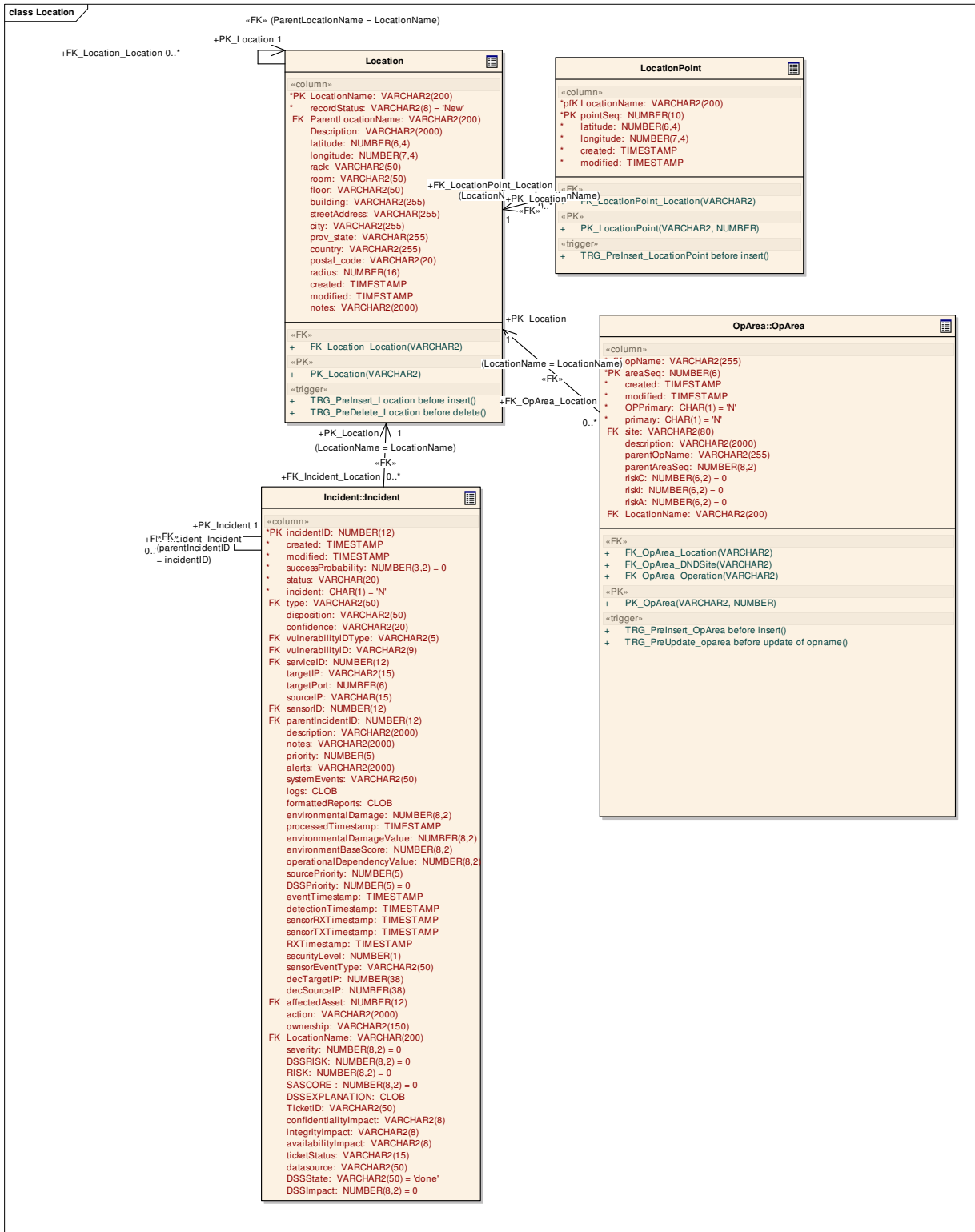
Connector	Source	Target
<u>Association</u> (IncidentType = type) Source -> Destination	Incident:IncidentTypeDocu ment	Incident:IncidentType

Operations

Method	Parameter
FK_IncidentTypeDo_IncidentType()	<u>VARCHAR2</u> IncidentType [in]
PK_IncidentTypeDocument()	<u>VARCHAR2</u> IncidentType [in] <u>VARCHAR2</u> DocumentName [in]

3.5.4.5.5 Location

Location - (Logical diagram)



3.5.4.5.5.1 Location

Type: **Class**

Stereotype: «table»

Notes:

Locations point to an exact latitude and longitude and are used to map JNDMS data onto our GeoSpatial views. Extra information can be used to help a person locate an asset, or an operational object when more detail than mapping information is needed. Locations can also link to an individual POC, or even a general indication for a network zone.

Attributes

Name	Type	Length	Precision	Scale	Notes
LocationName	VARCHAR2	200			A unique description of the location
recordStatus	VARCHAR2	8			Captures the current status of the record for internal JNDMS data verification. Allowable values include: New - a new record in teh database, to be verified. Pending - a record that is currently being reviewed or verified. Approved - A record that has been approved for us in the JNDMS. denied - A record that is not to be used in the JNDMS but is being retained.
ParentLocationName	VARCHAR2	200			The name of the location that this is contained within
Description	VARCHAR2	2000			An extended description of the location
latitude	NUMBER		6	4	The geospatial latitude of the location
longitude	NUMBER		7	4	The geospatial longitude of the location
rack	VARCHAR2	50			The rack identifier (if valid)

DN0678
Issue 4/2: 06 February 2012

Name	Type	Length	Precision	Scale	Notes
room	VARCHAR2	50			The room name/number of the location
floor	VARCHAR2	50			The floor name/number
building	VARCHAR2	255			A building identifier
streetAddress	VARCHAR	255			The street address of the location
city	VARCHAR2	255			The city in which the location appears
prov_state	VARCHAR	255			the provence or state of the location
country	VARCHAR2	255			The country for the location
postal_code	VARCHAR2	20			Postal code for the location
radius	NUMBER		16	0	The radius of the land area in meters
created	TIMESTAMP				Date the location entry was created
modified	TIMESTAMP				Date the location entry was last modified
notes	VARCHAR2	2000			Notes for the record. They may be entered by the operator or by the system.

Connections

Connector	Source	Target
<u>Association</u> (LocationName = LocationName) Source -> Destination	Point of Contact:POC	Location:Location
<u>Association</u> (LocationName = LocationName) Source -> Destination	Location:LocationPoint	Location:Location
<u>Association</u> (LocationName = LocationName) Source -> Destination	Incident:Incident	Location:Location

Connector	Source	Target
<u>Association</u> (LocationName = LocationName) Source -> Destination	OpArea:OpArea	Location:Location
<u>Association</u> (ParentLocationName = LocationName) Source -> Destination	Location:Location	Location:Location
<u>Association</u> (LocationName = LocationName) Source -> Destination	Topology:ZoneLocations	Location:Location

Operations

Method	Parameter
FK_Location_Location()	<u>VARCHAR2</u> ParentLocationName [in]
PK_Location()	<u>VARCHAR2</u> LocationName [in]
TRG_PreInsert_Location before insert()	
TRG_PreDelete_Location before delete()	

3.5.4.5.2 LocationPoint

Type: **Class**

Stereotype: «table»

Notes:

The Location can be defined by a series of points that will define a closed polygon. Note that if a single point is listed the location will be a point. 2 points will define a line. 3 or more points will define a closed polygon.

Attributes

Name	Type	Length	Precision	Scale	Notes
LocationName	VARCHAR2	200			Location name.
pointSeq	NUMBER		10	0	Points are held in sequence. The last point will link at the first point to close the polygon.
latitude	NUMBER	0	6	4	-90 - 90 stored as decimal degrees.
longitude	NUMBER	0	7	4	-180 - 180 stored as decimal degrees.
created	TIMESTAMP	0	0	0	Timestamp when record was created.
modified	TIMESTAMP	0	0	0	Timestamp of when the record was last modified.

Connections

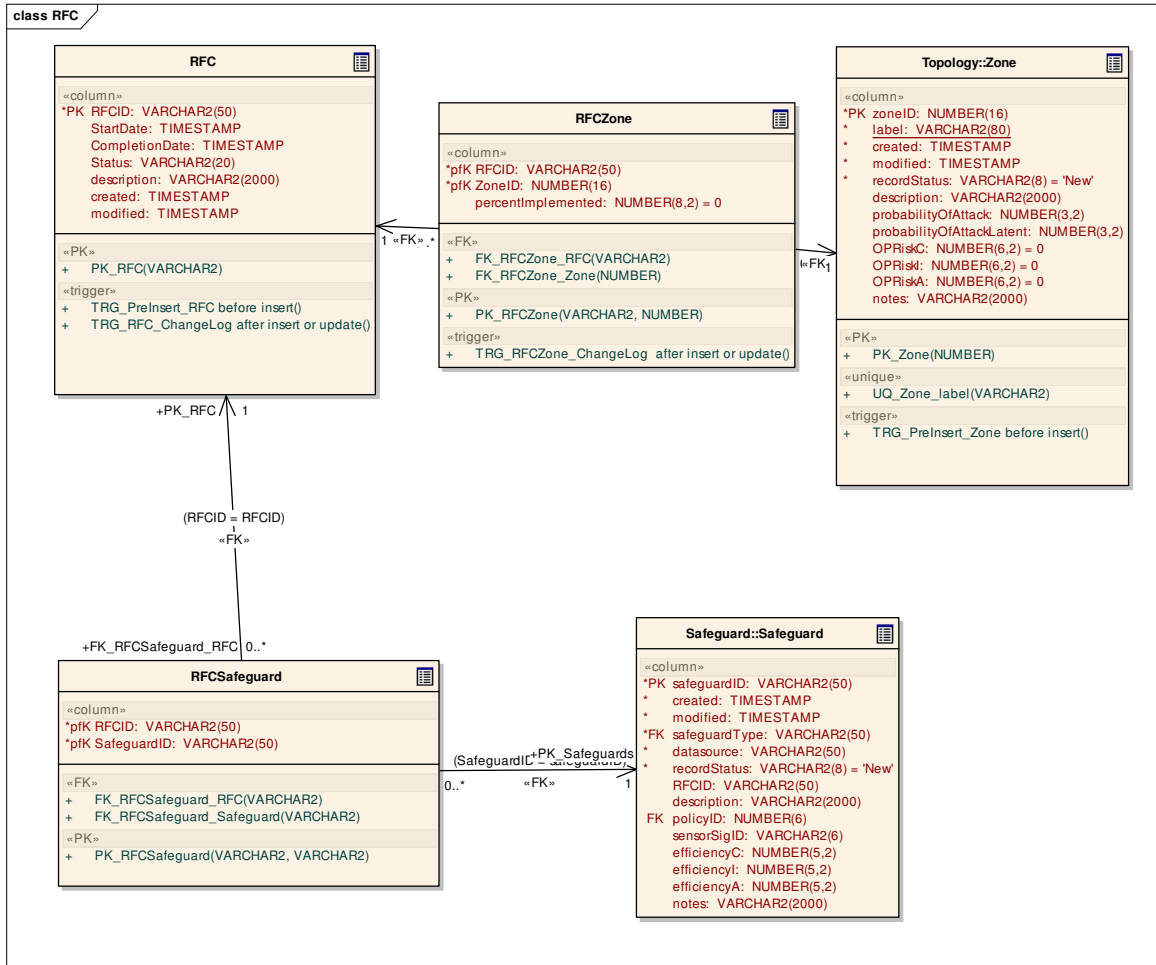
Connector	Source	Target
<u>Association</u> (LocationName = LocationName) Source -> Destination	Location:LocationPoint	Location:Location

Operations

Method	Parameter
FK_LocationPoint_Location()	<u>VARCHAR2</u> LocationName [in]
PK_LocationPoint()	<u>VARCHAR2</u> LocationName [in] <u>NUMBER</u> pointSeq [in]
TRG_PreInsert_LocationPoint before insert()	

3.5.4.5.6 RFC

RFC - (Logical diagram)



3.5.4.5.6.1 RFC

Type: **Class**

Stereotype: «table»

Notes:

Request For Changes (RFCs) are done via external methods. Within JNDMS we track the RFCs based on their external ID, and allow the users to manually track which Zones are affected by any one RFC. The change request will implement a safeguard and block all vulnerabilities on a given zone all at once.

Attributes

Name	Type	Length	Precision	Scale	Notes
RFCID	VARCHAR2	50			The DND ID of the RFC - possibly not numeric
StartDate	TIMESTAMP				Start date of the implementation of the RFC
CompletionDate	TIMESTAMP				The expected completion, or actual completion date (depending on the status)
Status	VARCHAR2	20			The current status of the RFC: New / In Progress / Implemented
description	VARCHAR2	2000			A descriptoin of the RFC
created	TIMESTAMP				The date the entry was created in the database
modified	TIMESTAMP				The date the entry was last modified in the database

Connections

Connector	Source	Target
<u>Association</u> (RFCID = RFCID) Source -> Destination	RFC:RFCZone	RFC:RFC
<u>Association</u> (RFCID = RFCID) Source -> Destination	RFC:RFCsafeguard	RFC:RFC

Operations

Method	Parameter
PK_RFC()	<u>VARCHAR2</u> RFCID [in]
TRG_PreInsert_RFC before insert()	
TRG_RFC_ChangeLog after insert or update()	

3.5.4.5.6.2 RFCSafeguard

Type: **Class**

Stereotype: «table»

Notes:

Table that links RFCs to Safeguards

Attributes

Name	Type	Length	Precision	Scale	Notes
RFCID	VARCHAR2	50			Foreign Key to RFC table
SafeguardID	VARCHAR2	50			Foreign Key to Safeguard table

Connections

Connector	Source	Target
<u>Association</u> (SafeguardID = safeguardID) Source -> Destination	RFC:RFCSafeguard	Safeguard:Safeguard
<u>Association</u> (RFCID = RFCID) Source -> Destination	RFC:RFCSafeguard	RFC:RFC

Operations

Method	Parameter
FK_RFCSafeguard_RFC()	<u>VARCHAR2</u> RFCID [in]
FK_RFCSafeguard_Safeguard()	<u>VARCHAR2</u> SafeguardID [in]
PK_RFCSafeguard()	<u>VARCHAR2</u> RFCID [in] <u>VARCHAR2</u> SafeguardID [in]

3.5.4.5.6.3 RFCZone

Type: **Class**

Stereotype: «*table*»

Notes:

The RFCZone links between RFCs and any one zone. This implies that a single change request can be for one, or more zones.

Attributes

Name	Type	Length	Precision	Scale	Notes
RFCID	VARCHAR2	50			Foreign Key to the RFC table
ZoneID	NUMBER		16	0	Foreign Key to the Zone Table
percentImplemented	NUMBER		8	2	Percent of the RFC that has been implemented on this network

Connections

Connector	Source	Target
<u>Association</u> (RFCID = RFCID) Source -> Destination	RFC:RFCZone	RFC:RFC
<u>Association</u> (ZoneID = zoneID) Source -> Destination	RFC:RFCZone	Topology:Zone

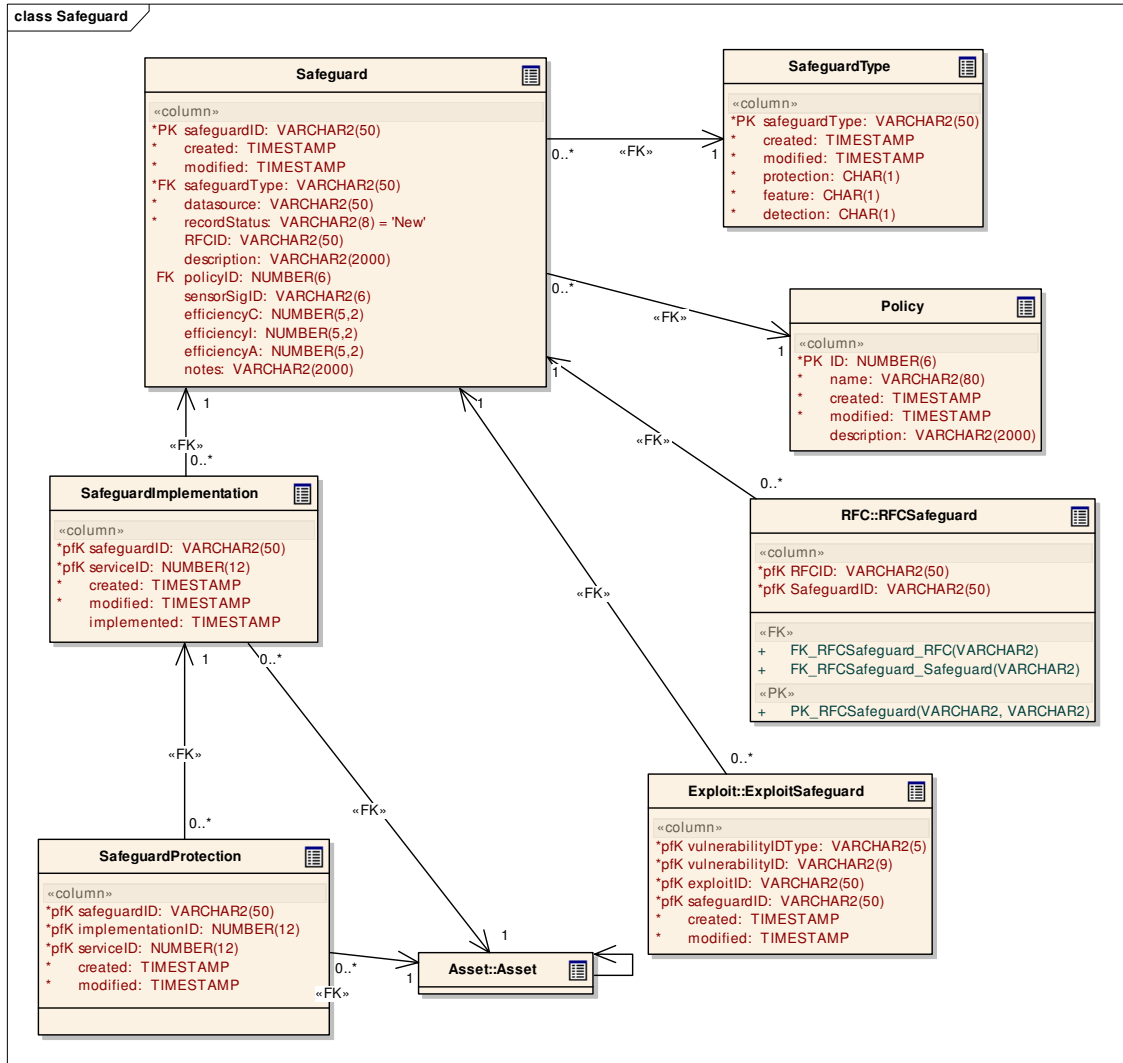
Operations

Method	Parameter
FK_RFCZone_RFC()	<u>VARCHAR2</u> RFCID [in]
FK_RFCZone_Zone()	<u>NUMBER</u> ZoneID [in]
PK_RFCZone()	<u>VARCHAR2</u> RFCID [in] <u>NUMBER</u> ZoneID [in]
TRG_RFCZone_ChangeLog after insert or update()	

3.5.4.5.7 Safeguard

The Safeguard package contains specific details on the entities related to safeguards.

Safeguard - (Logical diagram)



3.5.4.5.7.1 Policy

Type: **Class**

Stereotype: «table»

Notes:

The Policy table maintains a record of policies that have been set for the network(s). Safeguards may reference the policy to show that a given safeguard is an enforcement of the policy. Policies are statements only. There is no guarantee of enforcement of the policy.

Attributes

Name	Type	Length	Precision	Scale	Notes
ID	NUMBER	0	6	0	The ID is a unique artificial key for the policy.
name	VARCHAR2	80	0	0	The a name used to identify the policy.
created	TIMESTAMP	0	0	0	Timestamp when record was created.
modified	TIMESTAMP	0	0	0	Timestamp of last modification.
description	VARCHAR2	2000	0	0	Text description of the policy.

Connections

Connector	Source	Target
<u>Association</u> (policyID = ID) Source -> Destination	Safeguard:Safeguard	Safeguard:Policy

Operations

Method	Parameter
PK_Policy()	<u>NUMBER</u> ID [in]
TRG_PreInsert_Policy before insert()	

3.5.4.5.7.2 Safeguard

Type: **Class**

Stereotype: «table»

Notes:

A safeguard represents the mitigation against the risk of an incident or response to an incident. Safeguards are related to the asset that provides the safeguard (such as a firewall) through the safeguard implementation table. Initial safeguard data may be gleaned from the IT infrastructure (i.e., firewall policies, router access control lists, etc.)

A "Policy" table that would hold general security policies for assets has been considered. It has not been included at this time as there has not been a method identified that would verify the policy is enforced.

Attributes

Name	Type	Length	Precision	Scale	Notes
safeguardID	VARCHAR2	50	0	0	Safeguard ID is a unique name to describe the safeguard.
created	TIMESTAMP	0	0	0	Timestamp of record creation.
modified	TIMESTAMP	0	0	0	Timestamp of last modification.
safeguardType	VARCHAR2	50	0	0	The safeguard type.
datasource	VARCHAR2	50			Data source of safeguard.
recordStatus	VARCHAR2	8			Record status is internal JNDMS status value on the data. allowable values include: New - a new record. Pending - Record is under review. Approved - Record is approved for usage in the JNDMS. Denied - Record is not to be used in JNDMS but is to be retained.
RFCID	VARCHAR2	50	0	0	ID of the RFC for the safeguard.
description	VARCHAR2	2000	0	0	Free text description of the safeguard.
policyID	NUMBER	0	6	0	ID for the policy this safeguard is enforcing.
sensorSigID	VARCHAR2	6	0	0	Sensor signature ID.
efficiencyC	NUMBER	0	5	2	Efficiency of the Confidentiality of the safeguard.
efficiencyI	NUMBER	0	5	2	Efficiency of the Integrity of the

DN0678
Issue 4/2: 06 February 2012

Name	Type	Length	Precision	Scale	Notes
					Safeguard.
efficiencyA	NUMBER	0	5	2	Efficiency of the Availability of the Safeguard
notes	VARCHAR2	2000			Notes for the record entered by the operator or by the system.

Connections

Connector	Source	Target
<u>Association</u> (safeguardID = safeguardID) Source -> Destination	Exploit:ExploitSafeguard	Safeguard:Safeguard
<u>Association</u> (safeguardID = safeguardID) Source -> Destination	Safeguard:SafeguardImplementation	Safeguard:Safeguard
<u>Association</u> (policyID = ID) Source -> Destination	Safeguard:Safeguard	Safeguard:Policy
<u>Association</u> (safeguardType = safeguardType) Source -> Destination	Safeguard:Safeguard	Safeguard:SafeguardType
<u>Dependency</u> Source -> Destination	Data Transformation:Pre-Process Safeguard Data	Safeguard:Safeguard
<u>Association</u> (SafeguardID = safeguardID) Source -> Destination	RFC:RFCsSafeguard	Safeguard:Safeguard

Operations

Method	Parameter
FK_Safeguard_Policy()	<u>NUMBER</u> policyID [in]
FK_Safeguards_SafeguardType()	<u>VARCHAR2</u> safeguardType [in]
PK_Safeguards()	<u>VARCHAR2</u> safeguardID [in]
TRG_PreInsert_Safeguard before insert()	

3.5.4.5.7.3 SafeguardImplementation

Type: **Class**

Stereotype: «table»

Notes:

This table maps safeguards to the assets responsible to implement the safeguard. This asset may or may not be the asset that was vulnerable.

Attributes

Name	Type	Length	Precision	Scale	Notes
safeguardID	VARCHAR2	50	0	0	Sequence number for the safeguard.
serviceID	NUMBER	0	12	0	Asset service number.
created	TIMESTAMP	0	0	0	Timestamp record was created.
modified	TIMESTAMP	0	0	0	Timestamp record was last modified.
implemented	TIMESTAMP	0	0	0	Timestamp when the safeguard was implemented on the asset.

Connections

Connector	Source	Target
Association (safeguardID = safeguardID safeguardServiceID = serviceID) Source -> Destination	Topology:ZoneBorderSafeguard	Safeguard:SafeguardImplementation
Association (safeguardID = safeguardID) Source -> Destination	Safeguard:SafeguardImplementation	Safeguard:Safeguard
Association (safeguardID = safeguardID implementationID = serviceID) Source -> Destination	Safeguard:SafeguardProtection	Safeguard:SafeguardImplementation
Association (serviceID = serviceID) Source -> Destination	Safeguard:SafeguardImplementation	Asset:Asset

Operations

Method	Parameter
FK_SafeGuardApplication_Asset()	<u>NUMBER</u> serviceID [in]
FK_SafeguardImpleme_Safeguards()	<u>VARCHAR2</u> safeguardID [in]
PK_SafeGuardApplication()	<u>VARCHAR2</u> safeguardID [in] <u>NUMBER</u> serviceID [in]
TRG_PreInsert_SafeguardImpl before insert()	

3.5.4.5.7.4 SafeguardProtection

Type: **Class**

Stereotype: «table»

Notes:

This table maps the safeguard to the vulnerability instances that it is protecting.

Attributes

Name	Type	Length	Precision	Scale	Notes
safeguardID	VARCHAR2	50	0	0	Sequence number for the safeguard.
implementationID	NUMBER	0	12	0	The service ID of the asset implementing the protection..
serviceID	NUMBER		12	0	Asset service number.
created	TIMESTAMP	0	0	0	Timestamp record was created.
modified	TIMESTAMP	0	0	0	Timestamp of last modification.

Connections

Connector	Source	Target
<u>Association</u> (safeguardID = safeguardID implementationID = serviceID) Source -> Destination	Safeguard:SafeguardProtection	Safeguard:SafeguardImplementation
<u>Association</u> (serviceID = serviceID) Source -> Destination	Safeguard:SafeguardProtection	Asset:Asset

Operations

Method	Parameter
FK_SafeguardProtection_Asset()	<u>NUMBER</u> serviceID [in]
FK_SafeguardProt_SafeguardImpl()	<u>VARCHAR2</u> safeguardID [in] <u>NUMBER</u> implementationID [in]
PK_SafeguardProtection()	<u>VARCHAR2</u> safeguardID [in] <u>NUMBER</u> implementationID [in] <u>NUMBER</u> serviceID [in]
TRG_PreInsert_SafeguardProt before insert()	

3.5.4.5.7.5 SafeguardType

Type: **Class**

Stereotype: «table»

Notes:

Safeguard type is a look-up of safeguard types and their properties.

Attributes

Name	Type	Length	Precision	Scale	Notes
safeguardType	VARCHAR2	50	0	0	Type of safeguard.
created	TIMESTAMP	0	0	0	Timestamp of record creation.
modified	TIMESTAMP	0	0	0	Timestamp of last modification.
protection	CHAR	1	0	0	Does the safeguard offer protection against the incident. Y or N.
feature	CHAR	1	0	0	Is the safeguard a feature of the asset? Y or N
detection	CHAR	1	0	0	Does the safeguard detect the incident. Y or N.

Connections

Connector	Source	Target
<u>Association</u> (safeguardType = safeguardType) Source -> Destination	Safeguard:Safeguard	Safeguard:SafeguardType

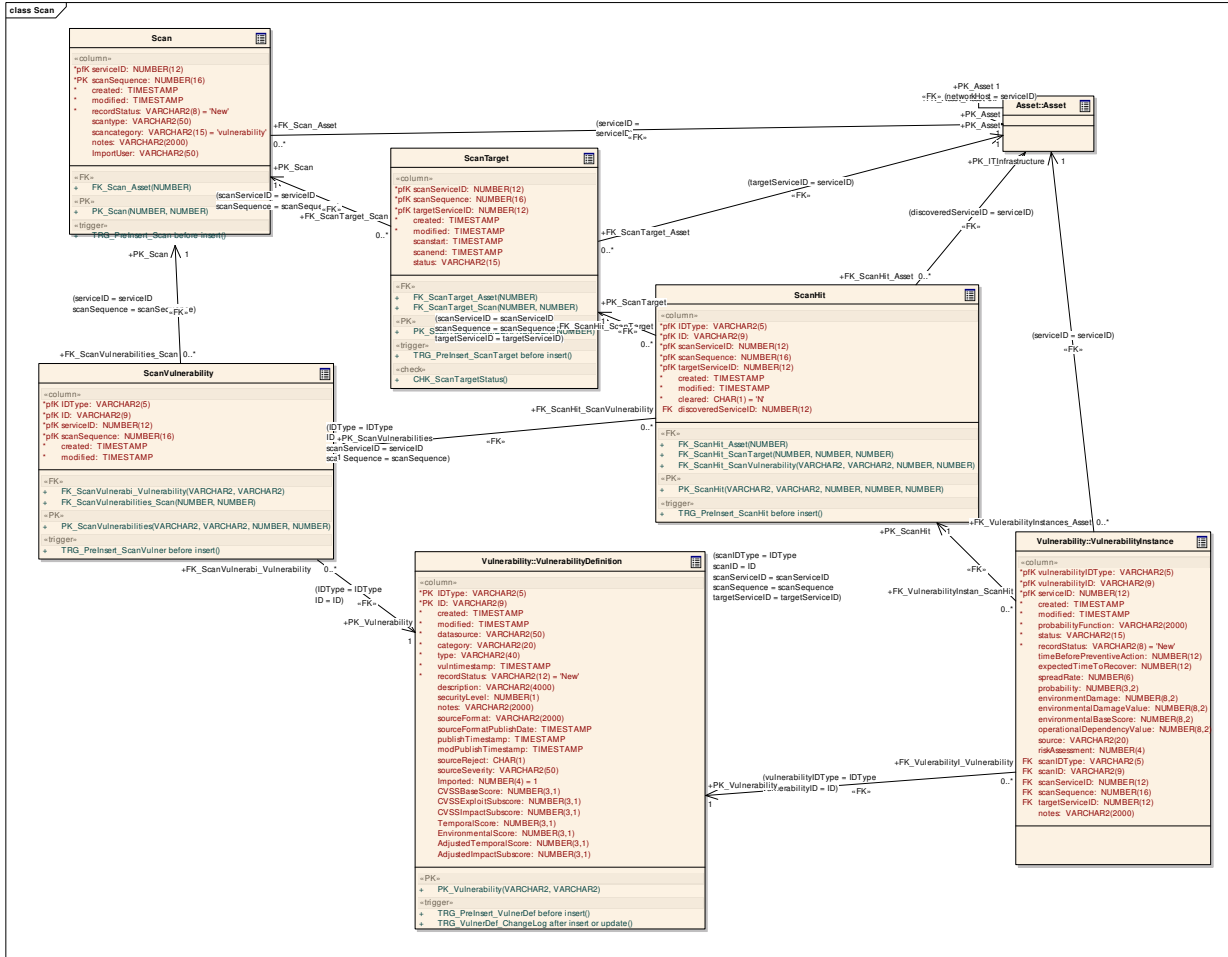
Operations

Method	Parameter
PK_SafeguardType()	<u>VARCHAR2</u> safeguardType [in]
TRG_PreInsert_SafeguardType before insert()	

3.5.4.5.8 Scan

This package holds tables related to performing scans.

Scan - (Logical diagram)



3.5.4.5.8.1 Scan

Type: **Class**

Stereotype: «table»

Notes:

A row in this table represents a single scan.

Attributes

Name	Type	Length	Precision	Scale	Notes
serviceID	NUMBER	0	12	0	Service ID of the scanner asset.
scanSequence	NUMBER	0	16	0	A sequence number representing the instance of the scan for a given scanner.
created	TIMESTAMP	0	0	0	Timestamp of record creation.
modified	TIMESTAMP	0	0	0	Timestamp of last modification
recordStatus	VARCHAR2	8			Record status indicate jndms internal status on the data. allowable values include: New - a new record. Pending - a record currently under review. Approved - A record that is to be used by the JNDMS. Denied - A record that is not to be used by the JNDMS but is retained.
scantype	VARCHAR2	50	0	0	Type of scan (local, remote,...).
scancategory	VARCHAR2	15			Categories of scans include: vulnerability, software, <td>
notes	VARCHAR2	2000			Notes on the record entered by teh operator or by the system.
ImportUser	VARCHAR2	50			The name of the user that performed the import

Connections

Connector	Source	Target
<u>Association</u> (scanServiceID = serviceID scanSequence = scanSequence) Source -> Destination	Scan:ScanTarget	Scan:Scan
<u>Association</u> (serviceID = serviceID scanSequence = scanSequence) Source -> Destination	Scan:ScanVulnerability	Scan:Scan
<u>Association</u> (serviceID = serviceID) Source -> Destination	Scan:Scan	Asset:Asset

Operations

Method	Parameter
FK_Scan_Asset()	<u>NUMBER</u> serviceID [in]
PK_Scan()	<u>NUMBER</u> serviceID [in] <u>NUMBER</u> scanSequence [in]
TRG_PreInsert_Scan before insert()	

3.5.4.5.8.2 ScanHit

Type: **Class**

Stereotype: «table»

Notes:

The list of vulnerabilities found on the scan target. The list may include 0 or more records.

Attributes

Name	Type	Length	Precision	Scale	Notes
IDType	VARCHAR2	5			Vulnerability ID Type.
ID	VARCHAR2	9	0	0	Vulnerability ID.
scanServiceID	NUMBER	0	12	0	Service ID for the scanner.
scanSequence	NUMBER	0	16	0	Sequence number of the scan for that scanner.
targetServiceID	NUMBER	0	12	0	Service ID of the asset that was found to have the vulnerability.
created	TIMESTAMP	0	0	0	Timestamp of record creation
modified	TIMESTAMP	0	0	0	Timestamp of last modification.
cleared	CHAR	1	0	0	'Y' indicates that a previous scanhit has been cleared by the same scanner source. 'N' indicates that a vulnerability instance has been detected.
discoveredService ID	NUMBER		12	0	The serviceID of the software asset that was discovered in the Hit. This doesn't apply to vulnerability scans - only software detection scans.

Connections

Connector	Source	Target
<u>Association</u> (scanServiceID = scanServiceID scanSequence = scanSequence targetServiceID = targetServiceID) Source -> Destination	Scan:ScanHit	Scan:ScanTarget
<u>Association</u> (scanIDType = IDType scanID = ID)	Vulnerability:VulnerabilityInstance	Scan:ScanHit

Connector	Source	Target
scanServiceID = scanServiceID scanSequence = scanSequence targetServiceID = targetServiceID Source -> Destination		
Association (IDType = IDType ID = ID scanServiceID = serviceID scanSequence = scanSequence) Source -> Destination	Scan:ScanHit	Scan:ScanVulnerability
Association (discoveredServiceID = serviceID) Source -> Destination	Scan:ScanHit	Asset:Asset

Operations

Method	Parameter
FK_ScanHit_Asset()	NUMBER discoveredServiceID [in]
FK_ScanHit_ScanTarget()	NUMBER scanServiceID [in] NUMBER scanSequence [in] NUMBER targetServiceID [in]
FK_ScanHit_ScanVulnerability()	VARCHAR2 IDType [in] VARCHAR2 ID [in] NUMBER scanServiceID [in] NUMBER scanSequence [in]
PK_ScanHit()	VARCHAR2 IDType [in] VARCHAR2 ID [in] NUMBER scanServiceID [in] NUMBER scanSequence [in] NUMBER targetServiceID [in]
TRG_PreInsert_ScanHit before insert()	

3.5.4.5.8.3 ScanTarget

Type: **Class**

Stereotype: «table»

Notes:

This table is an intersection table that will list the assets that were the target of a scan.

Attributes

Name	Type	Length	Precision	Scale	Notes
scanServiceID	NUMBER	0	12	0	Service ID of the Scanner.
scanSequence	NUMBER	0	16	0	Scan sequence for the scanner.
targetServiceID	NUMBER	0	12	0	Service ID of the asset scanned.
created	TIMESTAMP	0	0	0	Timestamp of record creation.
modified	TIMESTAMP	0	0	0	Timestamp of last modification.
scanstart	TIMESTAMP	0	0	0	Timestamp scan started.
scanend	TIMESTAMP	0	0	0	Timestamp when scan finished.
status	VARCHAR2	15	0	0	in progress pending complete failed

Connections

Connector	Source	Target
<u>Association</u> (scanServiceID = serviceID scanSequence = scanSequence) Source -> Destination	Scan:ScanTarget	Scan:Scan
<u>Association</u> (scanServiceID = scanServiceID scanSequence = scanSequence)	Scan:ScanHit	Scan:ScanTarget

Connector	Source	Target
targetServiceID = targetServiceID) Source -> Destination		
<u>Association</u> (targetServiceID = serviceID) Source -> Destination	Scan:ScanTarget	Asset:Asset

Operations

Method	Parameter
FK_ScanTarget_Asset()	<u>NUMBER</u> targetServiceID [in]
FK_ScanTarget_Scan()	<u>NUMBER</u> scanServiceID [in] <u>NUMBER</u> scanSequence [in]
PK_ScanTarget()	<u>NUMBER</u> scanServiceID [in] <u>NUMBER</u> scanSequence [in] <u>NUMBER</u> targetServiceID [in]
TRG_PreInsert_ScanTarget before insert()	
CHK_ScanTargetStatus()	

3.5.4.5.8.4 ScanVulnerability

Type: **Class**

Stereotype: «table»

Notes:

The list of vulnerabilities included in a scan.

Attributes

Name	Type	Length	Precision	Scale	Notes
IDType	VARCHAR2	5	0	0	Vulnerability ID Type. For example CVE.
ID	VARCHAR2	9	0	0	Vulnerability ID
serviceID	NUMBER	0	12	0	The Scanner asset service ID.
scanSequence	NUMBER	0	16	0	The sequence number of the scan for a given scanner.
created	TIMESTAMP	0	0	0	Timestamp of record creation.
modified	TIMESTAMP	0	0	0	Timestamp of last modification

Connections

Connector	Source	Target
Association (IDType = IDType ID = ID) Source -> Destination	Scan:ScanVulnerability	Vulnerability:VulnerabilityDef inition
Association (serviceID = serviceID scanSequence = scanSequence) Source -> Destination	Scan:ScanVulnerability	Scan:Scan
Association (IDType = IDType ID = ID scanServiceID = serviceID scanSequence = scanSequence) Source -> Destination	Scan:ScanHit	Scan:ScanVulnerability

Operations

Method	Parameter
FK_ScanVulnerabi_Vulnerability()	<u>VARCHAR2</u> IDType [in] <u>VARCHAR2</u> ID [in]
FK_ScanVulnerabilities_Scan()	<u>NUMBER</u> serviceID [in] <u>NUMBER</u> scanSequence [in]
PK_ScanVulnerabilities()	<u>VARCHAR2</u> IDType [in] <u>VARCHAR2</u> ID [in] <u>NUMBER</u> serviceID [in] <u>NUMBER</u> scanSequence [in]
TRG_PreInsert_ScanVulner before insert()	

3.5.4.5.9.1 VulnerabilityCaveat

Type: **Class**

Stereotype: «table»

Notes:

The countries (caveats) for the vulnerability definition.

Attributes

Name	Type	Length	Precision	Scale	Notes
IDType	VARCHAR2	5	0	0	Vulnerability ID type.
ID	VARCHAR2	9	0	0	The vulnerability ID.
caveat	VARCHAR2	3	0	0	Country code (caveat).
created	TIMESTAMP	0	0	0	Timestamp of record creation.
modified	TIMESTAMP	0	0	0	Timestamp of last modification.

Connections

Connector	Source	Target
<u>Association</u> (IDType = IDType ID = ID) Source -> Destination	Vulnerability:VulnerabilityCaveat	Vulnerability:VulnerabilityDefinit ion

Operations

Method	Parameter
FK_Vulnerability_Vulnerability()	<u>VARCHAR2</u> IDType [in] <u>VARCHAR2</u> ID [in]
PK_VulnerabilityDefinitionCave()	<u>VARCHAR2</u> IDType [in] <u>VARCHAR2</u> ID [in] <u>VARCHAR2</u> caveat [in]
TRG_PreInsert_VulnerCaveat before insert()	

3.5.4.5.9.2 VulnerabilityDefinition

Type: **Class**

Stereotype: «table»

Notes:

The vulnerability definition comes from the CVE, NVD plus other sources.

Definitions have a base score (risk).

The creation of a new vulnerability Definition will trigger vulnerability scans by JNDMS (based upon product information) as well as COTS vulnerability scanners.

Attributes

Name	Type	Length	Precision	Scale	Notes
IDType	VARCHAR2	5			Allowable values to include "CVE", "PHY", "NEW" (nvd.entry.type)
ID	VARCHAR2	9			ID (CVEID) needs to be modifiable. I suggest that for the purpose of the TD, modifiable is not required. In production, the data could be e "moved" to a new ID if necessary. (nvd.entry.seq)
created	TIMESTAMP	0	0	0	Timestamp record was created.
modified	TIMESTAMP	0	0	0	Timestamp of last modification.
datasource	VARCHAR2	50	0	0	Data source of vulnerability data.
category	VARCHAR2	20			Allowable Values of "Physical" and "Cyber".
type	VARCHAR2	40			Allowable values of weather, power, access OR application, database, operating system.
vulntimestamp	TIMESTAMP				Timestamp from vulnerability source. (nvd.entry.discovered)

DN0678
Issue 4/2: 06 February 2012

Name	Type	Length	Precision	Scale	Notes
recordStatus	VARCHAR2	12			<p>Status of the record internal to JNDMS. This field will be used by operators that manage the data to determine if any data requires their attention.</p> <p>New - New data that requires an operator to review and/or correct the data.</p> <p>Approved - Data that is approved for use by the JNDMS. Approval can be automated or manual.</p> <p>Pending - indicates that the record is under review.</p> <p>Denied - indicates that the record is not to be utilized by the JNDMS but is being retained. This may be useful to handle a report that gives a false record, on a regular basis.</p>
description	VARCHAR2	4000			<p>Free text vulnerability description.</p> <p>(nvd.entry.desc.descriptor nvd.entry.description)</p>
securityLevel	NUMBER	0	1	0	Security level of the vulnerability definition.
notes	VARCHAR2	2000			Notes about the record entered manually or by the system.
sourceFormat	VARCHAR2	2000			Information on the format of the data/document from the source. For example, nvd_xml_version. (nvd.nvd_xml_version)
sourceFormatPublishDate	TIMESTAMP				Date the source format was published. (nvd.pub_date)
publishTimestamp	TIMESTAMP				Timestamp vulnerability was published by the source. (nvd.entry.published)

DN0678
Issue 4/2: 06 February 2012

Name	Type	Length	Precision	Scale	Notes
modPublishTimestamp	TIMESTAMP				Timestamp of latest published modification. (nvd.entry.modified).
sourceReject	CHAR	1			indicates that this vulnerability entry has been rejected by the source. (nvd.entry.reject)
sourceSeverity	VARCHAR2	50			severity of the vulnerability as reported by the source. (nvd.entry.severity)
Imported	NUMBER		4	0	Whether or not the vulnerability was imported
CVSSBaseScore	NUMBER		3	1	Max CVSS version 2 Base Score of all related exploits
CVSSExploitSubscore	NUMBER		3	1	Max exploit subscore of all related exploits
CVSSImpactSubscore	NUMBER		3	1	Max impact subscore of all related exploits
TemporalScore	NUMBER		3	1	Max temporal score of all related exploits
EnvironmentalScore	NUMBER		3	1	Max environmental score of all related exploits
AdjustedTemporalScore	NUMBER		3	1	Max adjusted temporal score of all related exploits
AdjustedImpactSubscore	NUMBER		3	1	Max adjusted impact subscore of all related exploits

Connections

Connector	Source	Target
<u>Association</u> (vulnerabilityIDType = IDType vulnerabilityID = ID) Source -> Destination	Exploit:Exploit	Vulnerability:VulnerabilityDefinition
<u>Dependency</u> Source -> Destination	Data Transformation:Pre-Process Vulnerability and Exploit Data	Vulnerability:VulnerabilityDefinition
<u>Dependency</u> Source -> Destination	Data Warehousing:Store Vulnerability Definitions	Vulnerability:VulnerabilityDefinition
<u>Dependency</u> Source -> Destination	Security Information Management:Correlate Security Events and Vulnerability Scans	Vulnerability:VulnerabilityDefinition
<u>Dependency</u> Source -> Destination	Data Warehousing:Store Vulnerability and Exploit Data	Vulnerability:VulnerabilityDefinition
<u>Association</u> (IDType = IDType ID = ID) Source -> Destination	Scan:ScanVulnerability	Vulnerability:VulnerabilityDefinition
<u>Association</u> (vulnerabilityIDType = IDType vulnerabilityID = ID) Source -> Destination	Vulnerability:VulnerabilityReference	Vulnerability:VulnerabilityDefinition
<u>Association</u> (IDType = IDType ID = ID) Source -> Destination	Vulnerability:VulnerabilityProduct	Vulnerability:VulnerabilityDefinition

Connector	Source	Target
Association (IDType = IDType ID = ID) Source -> Destination	Vulnerability:VulnerabilityCa veat	Vulnerability:VulnerabilityDef inition
Association (vulnerabilityIDType = IDType vulnerabilityID = ID) Source -> Destination	Vulnerability:VulnerabilityInst ance	Vulnerability:VulnerabilityDef inition
Association (IDType = IDType ID = ID) Source -> Destination	Vulnerability:VulnerabilityDef initionLink	Vulnerability:VulnerabilityDef inition
Association (LinkIDType = IDType LinkID = ID) Source -> Destination	Vulnerability:VulnerabilityDef initionLink	Vulnerability:VulnerabilityDef inition
Association Unspecified	dana:UI Control	Vulnerability:VulnerabilityDef inition

Operations

Method	Parameter
PK_Vulnerability()	<u>VARCHAR2</u> IDType [in] <u>VARCHAR2</u> ID [in]
TRG_PreInsert_VulnerDef before insert()	
TRG_VulnerDef_ChangeLog after insert or update()	

3.5.4.5.9.3 VulnerabilityDefinitionLink

Type: **Class**

Stereotype: «table»

Notes:

This is a linking table that allows vulnerability definitions to be associated with one or more other vulnerability definitions. This comes into play with NVAT users editing custom vulnerabilities based on custom sources. (Possibly internal software that doesn't report to Vulnerability Definition companies)

Attributes

Name	Type	Length	Precision	Scale	Notes
IDType	VARCHAR2	5			Foreign Key to the VulnerabilityDefinition table
ID	VARCHAR	9			Foreign Key to the VulnerabilityDefinition table
LinkIDType	VARCHAR2	5			Foreign Key to the VulnerabilityDefinition table specifying the linked Vulnerability
LinkID	VARCHAR2	9			Foreign Key to the VulnerabilityDefinition table specifying the linked Vulnerability
Branched	NUMBER		4	0	Whether or not the Linked item was branched from the IDType

Connections

Connector	Source	Target
Association (IDType = IDType ID = ID) Source -> Destination	Vulnerability:VulnerabilityDef initionLink	Vulnerability:VulnerabilityDef inition
Association (LinkIDType = IDType LinkID = ID) Source -> Destination	Vulnerability:VulnerabilityDef initionLink	Vulnerability:VulnerabilityDef inition

Operations

Method	Parameter
FK_Vulnerability_Vulnera_Link()	<u>VARCHAR</u> LinkIDType [in] <u>VARCHAR2</u> LinkID [in]
FK_Vulnerability_Vulnerability()	<u>VARCHAR2</u> IDType [in] <u>VARCHAR</u> ID [in]
FK_Vulnerability_VulnLink1()	<u>VARCHAR2</u> IDType [in] <u>VARCHAR</u> ID [in]
FK_Vulnerability_VulnLink2()	<u>VARCHAR2</u> LinkIDType [in] <u>VARCHAR2</u> LinkID [in]
PK_VulnerabilityDefinitionLink()	<u>VARCHAR2</u> IDType [in] <u>VARCHAR</u> ID [in] <u>VARCHAR2</u> LinkIDType [in] <u>VARCHAR2</u> LinkID [in]

3.5.4.5.9.4 VulnerabilityInstance

Type: **Class**

Stereotype: «table»

Notes:

Vulnerability instances link a vulnerability to specific assets known to the JNDMS. Through these assets, the risk to operations can be assessed as part of setting priorities for analyst intervention.

Attributes

Name	Type	Length	Precision	Scale	Notes
vulnerabilityIDType	VARCHAR2	5	0	0	Vulnerability ID type.
vulnerabilityID	VARCHAR2	9	0	0	Vulnerability ID.
serviceID	NUMBER	0	12	0	Service id of the asset.
created	TIMESTAMP	0	0	0	Timestamp of record creation.
modified	TIMESTAMP	0	0	0	Timestamp of last modification.
probabilityFunction	VARCHAR2	2000	0	0	This allows a probability function to be defined for the specific vulnerability instance.
status	VARCHAR2	15	0	0	Current status of the vulnerability instance. Active - Wild Active Mitigated Resolved Forecasted
recordStatus	VARCHAR2	8			Record status for internal JNDMS use. Allowable values include: New - A new record in the database. Pending - a record currently under review. Approved - A record that is to be used by the JNDMS. Denied - A record that is not to be used by the JNDMS but is to be retained.
timeBeforePreve	NUMBER	0	12	0	Count in seconds of the time the

DN0678
Issue 4/2: 06 February 2012

Name	Type	Length	Precision	Scale	Notes
preventiveAction					vulnerability/incident existed prior to preventive action was taken.
expectedTimeToRecover	NUMBER	0	12	0	Count in seconds of time required to recover from the vulnerability/incident.
spreadRate	NUMBER	0	6	0	Instances per minute.
probability	NUMBER	0	3	2	Probability of the risk.
environmentDamage	NUMBER	0	8	2	Value of the environmental damage.
environmentalDamageValue	NUMBER	0	8	2	Value of the environmental damage.
environmentalBaseScore	NUMBER	0	8	2	Base value for the environment.
operationalDependencyValue	NUMBER	0	8	2	Score of the operational dependency of the incident.
source	VARCHAR2	20	0	0	Source of the data. Incident DSS Operator Nessus Scanner
riskAssessment	NUMBER		4	0	The JNDMS DSS calculated risk assessment for the vulnerability instance. An analyst may override the calculated value.
scanIDType	VARCHAR2	5	0	0	Scan hit that created the vulnerability instance.
scanID	VARCHAR2	9	0	0	Scan hit that created the vulnerability instance.
scanServiceID	NUMBER	0	12	0	Scan hit that created the vulnerability instance.
scanSequence	NUMBER	0	16	0	Scan hit that created the vulnerability instance.
targetServiceID	NUMBER	0	12	0	Scan hit that created the vulnerability instance.
notes	VARCHAR2	2000			Notes on the record, entered by the operator or by the system.

Connections

Connector	Source	Target
<p><u>Association</u>(vulnerabilityIDType = vulnerabilityIDType vulnerabilityID = vulnerabilityID serviceID = serviceID) Source -> Destination</p>	<p>Incident:Incident</p>	<p>Vulnerability:VulnerabilityInstance</p>
<p><u>Association</u>(scanIDType = IDType scanID = ID scanServiceID = scanServiceID scanSequence = scanSequence targetServiceID = targetServiceID) Source -> Destination</p>	<p>Vulnerability:VulnerabilityInstance</p>	<p>Scan:ScanHit</p>
<p><u>Association</u>(vulnerabilityIDType = vulnerabilityIDType vulnerabilityID = vulnerabilityID serviceID = serviceID) Source -> Destination</p>	<p>Vulnerability:vulnerabilityInstancePort</p>	<p>Vulnerability:VulnerabilityInstance</p>
<p><u>Association</u>(vulnerabilityIDType = IDType vulnerabilityID = ID) Source -> Destination</p>	<p>Vulnerability:VulnerabilityInstance</p>	<p>Vulnerability:VulnerabilityDefinition</p>
<p><u>Dependency</u> Source -> Destination</p>	<p>Security Information Management:Pre-Process Security Events</p>	<p>Vulnerability:VulnerabilityInstance</p>
<p><u>Dependency</u> Source -> Destination</p>	<p>Data Warehousing:Store Vulnerability Instances</p>	<p>Vulnerability:VulnerabilityInstance</p>

Connector	Source	Target
<u>Dependency</u> Source -> Destination	Security Information Management:Correlate Security Events and Vulnerability Scans	Vulnerability:VulnerabilityInstance
<u>Association</u> (serviceID = serviceID) Source -> Destination	Vulnerability:VulnerabilityInstance	Asset:Asset

Operations

Method	Parameter
FK_VulnerabilityInstan_ScanHit()	<u>VARCHAR2</u> scanIDType [in] <u>VARCHAR2</u> scanID [in] <u>NUMBER</u> scanServiceID [in] <u>NUMBER</u> scanSequence [in] <u>NUMBER</u> targetServiceID [in]
FK_VulnerabilityI_Vulnerability()	<u>VARCHAR2</u> vulnerabilityIDType [in] <u>VARCHAR2</u> vulnerabilityID [in]
FK_VulnerabilityInstances_Asset()	<u>NUMBER</u> serviceID [in]
PK_VulnerabilityInstances()	<u>VARCHAR2</u> vulnerabilityIDType [in] <u>VARCHAR2</u> vulnerabilityID [in] <u>NUMBER</u> serviceID [in]
TRG_PreInsert_VulnerInst before insert()	
TRG_PreDelete_VulnerInst before delete()	

3.5.4.5.9.5 vulnerabilityInstancePort

Type: **Class**

Stereotype: «table»

Notes:

This contains the port, protocol pairs that are open that are open for a given vulnerability instance pair.

Attributes

Name	Type	Length	Precision	Scale	Notes
vulnerabilityIDType	VARCHAR2	5			ID Type
vulnerabilityID	VARCHAR2	9	0	0	Vulnerability ID.
serviceID	NUMBER	0	12	0	Service ID of the vulnerable asset.
port	NUMBER	0	6	0	Port number where vulnerability exists on this asset.
protocol	VARCHAR2	50	0	0	Protocol available on that port.
created	TIMESTAMP	0	0	0	Timestamp of record creation.
modified	TIMESTAMP	0	0	0	Timestamp of last modification

Connections

Connector	Source	Target
<u>Association</u> (vulnerabilityIDType = vulnerabilityIDType vulnerabilityID = vulnerabilityID serviceID = serviceID) Source -> Destination	Vulnerability:vulnerabilityInstancePort	Vulnerability:VulnerabilityInstance

Operations

Method	Parameter
FK_vulinst_vulinstport()	<u>VARCHAR2</u> vulnerabilityIDType [in] <u>VARCHAR2</u> vulnerabilityID [in] <u>NUMBER</u> serviceID [in]
PK_vulnerabilityInstancePort()	<u>VARCHAR2</u> vulnerabilityIDType [in] <u>VARCHAR2</u> vulnerabilityID [in] <u>NUMBER</u> serviceID [in] <u>NUMBER</u> port [in] <u>VARCHAR2</u> protocol [in]
TRG_PreInsert_VullInstPort before insert()	

3.5.4.5.9.6 VulnerabilityProduct

Type: **Class**

Stereotype: «table»

Notes:

This table is the list of products affect by the vulnerability.

Attributes

Name	Type	Length	Precision	Scale	Notes
IDType	VARCHAR2	5	0	0	Vulnerability ID Type.
ID	VARCHAR2	9	0	0	Vulnerability ID.
vendor	VARCHAR2	255	0	0	Vendor
product	VARCHAR2	255	0	0	Product
version	VARCHAR2	255	0	0	Version
created	TIMESTAMP	0	0	0	Timestamp of record creation.
modified	TIMESTAMP	0	0	0	Timestamp of last modification.
secondaryVendor	VARCHAR2	50			An O.S. that may also be included
secondaryProduct	VARCHAR2	50			An O.S. that may also be included
secondaryVersion	VARCHAR2	50			An O.S. that may also be included

Connections

Connector	Source	Target
Association (IDType = IDType ID = ID) Source -> Destination	Vulnerability:VulnerabilityPro duct	Vulnerability:VulnerabilityDef inition
Association (secondaryVendor = vendor secondaryProduct = product secondaryVersion = version) Source -> Destination	Vulnerability:VulnerabilityPro duct	Product:Product

Connector	Source	Target
Association (vendor = vendor product = product version = version) Source -> Destination	Vulnerability:VulnerabilityProduct	Product:Product

Operations

Method	Parameter
FK_VulnerabilityProduct_Product()	<u>VARCHAR2</u> secondaryVendor [in] <u>VARCHAR2</u> secondaryProduct [in] <u>VARCHAR2</u> secondaryVersion [in]
FK_Vulner_VulnerProd()	<u>VARCHAR2</u> IDType [in] <u>VARCHAR2</u> ID [in]
FK_VulnerProd_Product()	<u>VARCHAR2</u> vendor [in] <u>VARCHAR2</u> product [in] <u>VARCHAR2</u> version [in]
PK_VulnerabilityProduct()	<u>VARCHAR2</u> IDType [in] <u>VARCHAR2</u> ID [in] <u>VARCHAR2</u> vendor [in] <u>VARCHAR2</u> product [in] <u>VARCHAR2</u> version [in]
TRG_PreInsert_VulnerProd before insert()	

3.5.4.5.9.7 VulnerabilityReference

Type: **Class**

Stereotype: «table»

Notes:

Additional information on the vulnerability.

Attributes

Name	Type	Length	Precision	Scale	Notes
vulnerabilityIDType	VARCHAR2	5	0	0	Vulnerability ID type.
vulnerabilityID	VARCHAR2	9	0	0	Vulnerability ID.
referenceSeq	NUMBER	0	6	0	Sequence number of reference.
created	TIMESTAMP	0	0	0	Timestamp record was created.
modified	TIMESTAMP	0	0	0	Timestamp of the last modification.
type	VARCHAR2	50			Store the type of reference: 'Patch Information'
URL	VARCHAR2	2000	0	0	Links to other sources of information on the vulnerability.
text	VARCHAR2	1000			Copied text.

Connections

Connector	Source	Target
Association (vulnerabilityIDType = IDType vulnerabilityID = ID) Source -> Destination	Vulnerability:VulnerabilityReference	Vulnerability:VulnerabilityDefinition

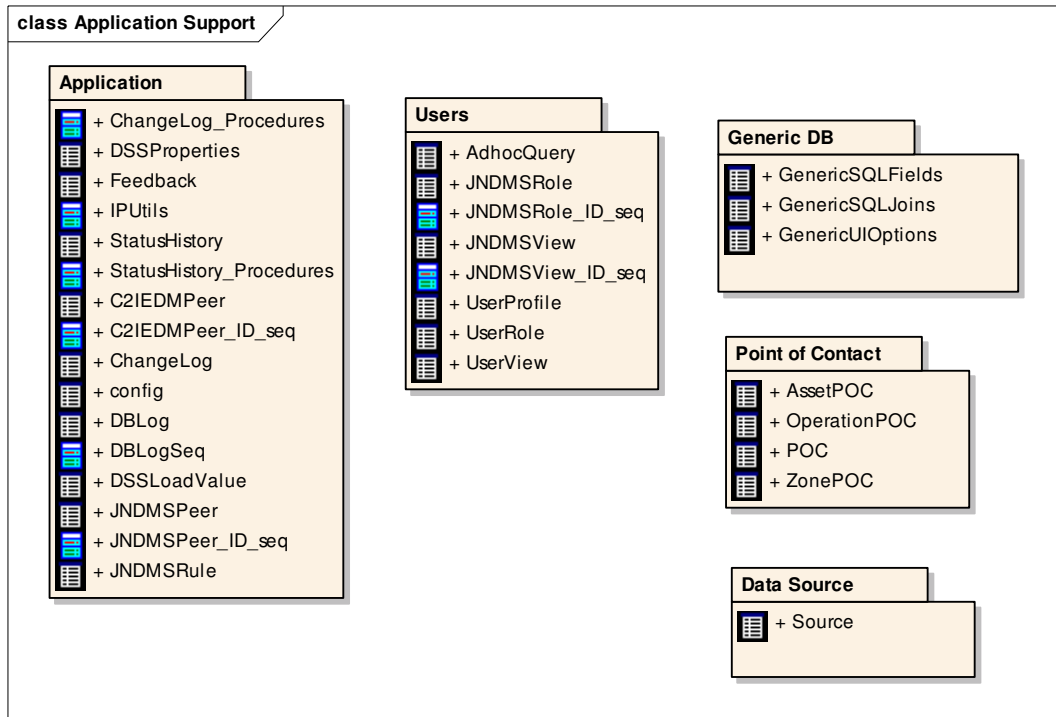
Operations

Method	Parameter
PK_references()	VARCHAR2 vulnerabilityIDType [in] VARCHAR2 vulnerabilityID [in] NUMBER referenceSeq [in]
PK_vulnerability_reference()	VARCHAR2 vulnerabilityIDType [in] VARCHAR2 vulnerabilityID [in]
TRG_PreInsert_VulnerRef before insert()	

3.5.4.5.10 Application Support

The application support package contains data elements outside of the data handled by the JNDMS (i.e., network security application data). This package supports the JNDMS application itself (i.e., users, application configuration, database connection strings, etc.).

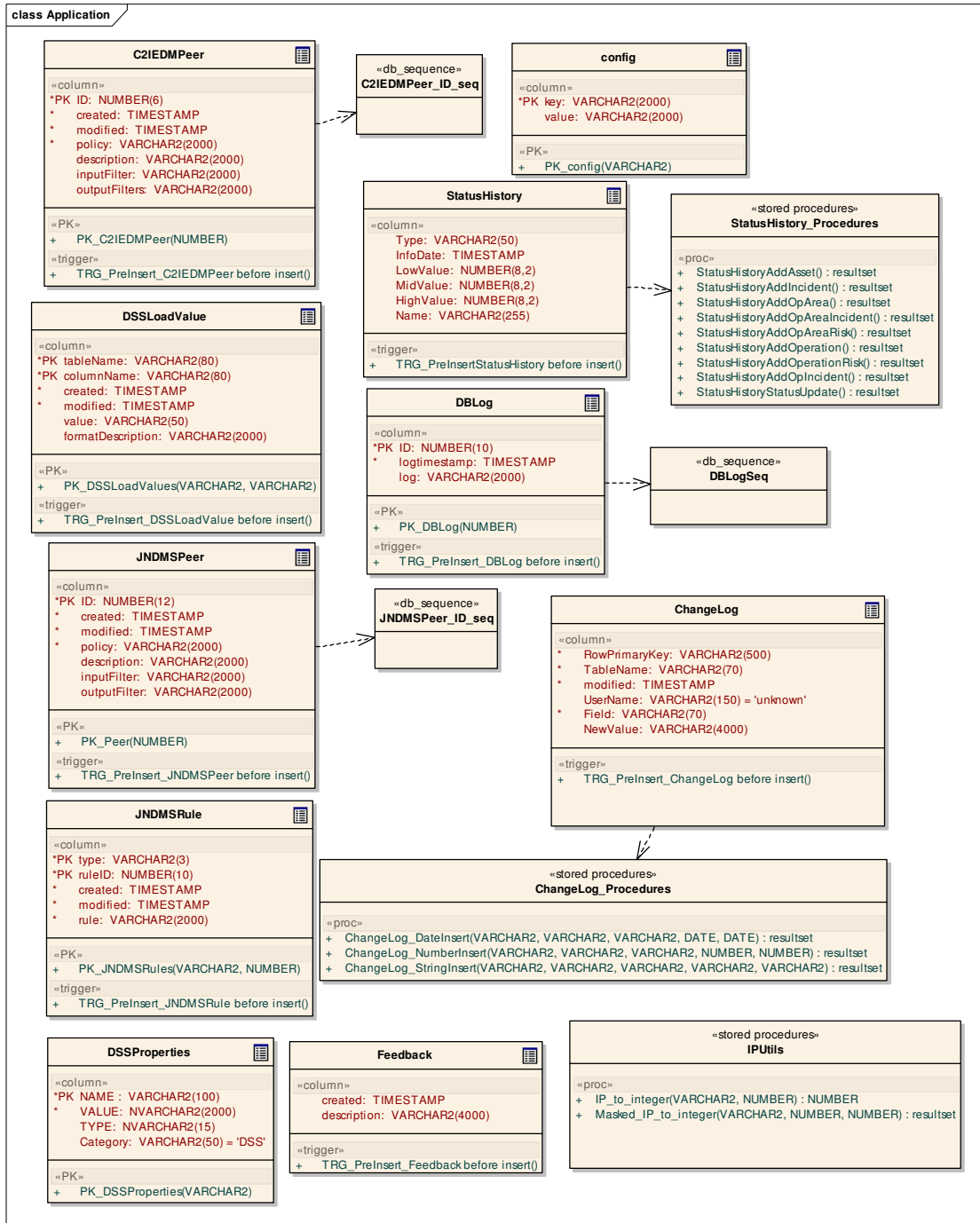
Application Support - (Logical diagram)



3.5.4.5.10.1 Application

The Application package contains elements that will be stored in the database that support generic application configuration.

Application - (Logical diagram)



3.5.4.5.10.2 C2IEDMPeer

Type: **Class**

Stereotype: «table»

Notes:

Attributes

Name	Type	Length	Precision	Scale	Notes
ID	NUMBER	0	6	0	Identifier for C2IEDM/JC3IEDM peer system.
created	TIMESTAMP	0	0	0	Timestamp record was created.
modified	TIMESTAMP	0	0	0	Timestamp of last modification.
policy	VARCHAR2	2000	0	0	Policy definition for data exchange.
description	VARCHAR2	2000	0	0	Description of the peer system.
inputFilter	VARCHAR2	2000	0	0	Filter definition(s) for incoming data.
outputFilters	VARCHAR2	2000	0	0	Output filter definition(s) for data exchange.

Connections

Connector	Source	Target
Dependency Source -> Destination	Situational Awareness Data Sharing: Working with a Coalition Partner	Application:C2IEDMPeer
Dependency Source -> Destination	Application:C2IEDMPeer	Application:C2IEDMPeer_ID_seq

Operations

Method	Parameter
PK_C2IEDMPeer()	<u>NUMBER</u> ID [in]
TRG_PreInsert_C2IEDMPeer before insert()	

3.5.4.5.10.3 C2IEDMPeer_ID_seq

Type: **Class**

Stereotype: «db_sequence»

Notes:

This is a sequence generator to provide a unique identifier for C2IEDM Peer records.

Connections

Connector	Source	Target
<u>Dependency</u> Source -> Destination	Application:C2IEDMPeer	Application:C2IEDMPeer_ID_seq

3.5.4.5.10.4 ChangeLog

Type: **Class**

Stereotype: «table»

Notes:

The ChangeLog will track updates to various tables in a way that can be used to track the historical context of information.

A primary key is not set on this table. This is due to the fact that it would include most fields from the source table (i.e., where the change originated), and thus the index would be almost as large as the table itself. Once we determine how we will use the table, the indexes should be defined.

One option is to implement it as an index only table (IOT).

Attributes

Name	Type	Length	Precision	Scale	Notes
RowPrimaryKey	VARCHAR2	500	0	0	A text version of the field(s) that comprise the primary key from the table that was updated. This is an underscore separated list of integer / text values that combine to define the primary key. They are to be listed in the order the attributes are defined in the class diagram (same as the order the columns appear in the table)
TableName	VARCHAR2	70	0	0	The table that contained the change
modified	TIMESTAMP	0	0	0	The time of the change
UserName	VARCHAR2	150			Name of the user that made the modification
Field	VARCHAR2	70	0	0	The name of the field that was modified
NewValue	VARCHAR2	4000			The string value of what the new value currently is. We can find the old value by looking at the history of the table.

Connections

Connector	Source	Target
<u>Dependency</u> Source -> Destination	Application:ChangeLog	Application:ChangeLog_Procedures

Operations

Method	Parameter
TRG_PreInsert_ChangeLog before insert()	

3.5.4.5.10.5 ChangeLog_Procedures

Type: **Class**

Stereotype: «stored procedures»

Notes:

Connections

Connector	Source	Target
Dependency Source -> Destination	Application:ChangeLog	Application:ChangeLog_Procedures

Operations

Method	Parameter
ChangeLog_DateInsert()	<u>VARCHAR2</u> 1_RowPrimaryKey [IN] <u>VARCHAR2</u> 1_TableName [IN] <u>VARCHAR2</u> 1_Field [IN] <u>DATE</u> 1_NewValue [IN] <u>DATE</u> 1_OldValue [IN]
ChangeLog_NumberInsert()	<u>VARCHAR2</u> 1_RowPrimaryKey [IN] <u>VARCHAR2</u> 1_TableName [IN] <u>VARCHAR2</u> 1_Field [IN] <u>NUMBER</u> 1_NewValue [IN] <u>NUMBER</u> 1_OldValue [IN]
ChangeLog_StringInsert()	<u>VARCHAR2</u> 1_RowPrimaryKey [IN] <u>VARCHAR2</u> 1_TableName [IN] <u>VARCHAR2</u> 1_Field [IN] <u>VARCHAR2</u> 1_NewValue [IN] <u>VARCHAR2</u> 1_OldValue [IN]

3.5.4.5.10.6 config

Type: **Class**

Stereotype: «table»

Notes:

The configuration table allows various configuration values for the system to be stored and accessed via the database.

Attributes

Name	Type	Length	Precision	Scale	Notes
key	VARCHAR2	2000	0	0	This is the configuration item, the JNDMS will have to know to search for this key.
value	VARCHAR2	2000	0	0	The value assigned to the key for this JNDMS instance.

Operations

Method	Parameter
PK_config()	<u>VARCHAR2</u> key [in]

3.5.4.5.10.7 DBLog

Type: **Class**

Stereotype: «table»

Notes:

This table exists primarily for development and debugging purposes. It is not intended to be part of the core system but will be preserved to support further development and debugging of the JNDMS.

Attributes

Name	Type	Length	Precision	Scale	Notes
ID	NUMBER	0	10	0	Unique identifier for the log entry.
logtimestamp	TIMESTAMP	0	0	0	Timestamp of the log entry.
log	VARCHAR2	2000	0	0	Log entry text.

Connections

Connector	Source	Target
<u>Dependency</u> Source -> Destination	Application:DBLog	Application:DBLogSeq

Operations

Method	Parameter
PK_DBLog()	<u>NUMBER</u> ID [in]
TRG_PreInsert_DBLog before insert()	

3.5.4.5.10.8 DBLogSeq

Type: **Class**

Stereotype: «*db_sequence*»

Notes:

CREATE SEQUENCE %className%

MINVALUE 1

START WITH 1

INCREMENT BY 1

NOCYCLE

NOCACHE;

Connections

Connector	Source	Target
<u>Dependency</u> Source -> Destination	Application:DBLog	Application:DBLogSeq

3.5.4.5.10.9 DSSLoadValue

Type: **Class**

Stereotype: «table»

Notes:

This table will keep initial load values to use for various attributes in the system. These values can be adjusted to try and "tune" the SA.

Attributes

Name	Type	Length	Precision	Scale	Notes
tableName	VARCHAR2	80	0	0	Database table.
columnName	VARCHAR2	80	0	0	Database table column name.
created	TIMESTAMP	0	0	0	Timestamp record was created.
modified	TIMESTAMP	0	0	0	Timestamp of last modification.
value	VARCHAR2	50	0	0	Value to be used for initial load. Modeled as text for flexibility. Value should be verified against the column format description.
formatDescription	VARCHAR2	2000	0	0	Describe the format and acceptable ranges for the column.

Operations

Method	Parameter
PK_DSSLoadValues()	<u>VARCHAR2</u> tableName [in]
	<u>VARCHAR2</u> columnName [in]
TRG_PreInsert_DSSLoadValue before insert()	

3.5.4.5.10.10 DSSProperties

Type: **Class**

Stereotype: «table»

Notes:

New table to store user modifiable DSS parameters:

Attributes

Name	Type	Length	Precision	Scale	Notes
NAME	VARCHAR2	100			primary key, name of property
VALUE	NVARCHAR2	2000			value of property
TYPE	NVARCHAR2	15			specifies data type of value (i.e. string, long, etc...) DSS assumes NULL = string
Category	VARCHAR2	50			Store the category of the property. Most are DSS - some are not

Operations

Method	Parameter
PK_DSSProperties()	<u>VARCHAR2</u> NAME [in]

3.5.4.5.10.11 Feedback

Type: **Class**

Stereotype: «table»

Notes:

User feedback is captured as simple strings in this table.

Attributes

Name	Type	Length	Precision	Scale	Notes
created	TIMESTAMP				The date the feedback was entered into the database
description	VARCHAR2	4000			The description of the feedback item.

Operations

Method	Parameter
TRG_PreInsert_Feedback before insert()	

3.5.4.5.10.12 IPUtils

Type: **Class**

Stereotype: «*stored procedures*»

Notes:

Functions to perform the mapping from a string IP address to a decimal IP address. This takes the requirement off the JSS, and allows queries on the data to perform consistently.

Operations

Method	Parameter
IP_to_integer()	<u>VARCHAR2</u> IP [IN] <u>NUMBER</u> decIP [OUT]
Masked_IP_to_integer()	<u>VARCHAR2</u> IP [IN] <u>NUMBER</u> startDecIP [OUT] <u>NUMBER</u> endDecIP [OUT]

3.5.4.5.10.13 JNDMSPeer

Type: **Class**

Stereotype: «table»

Notes:

The JNDMS Peer entity contains data required to manage the data exchange with peer JNDMS systems.

Attributes

Name	Type	Length	Precision	Scale	Notes
ID	NUMBER	0	12	0	Identified of JNDMS peer system.
created	TIMESTAMP	0	0	0	Timestamp record was created.
modified	TIMESTAMP	0	0	0	Timestamp of last modification.
policy	VARCHAR2	2000	0	0	Indicates the nature of the data exchange between peers.
description	VARCHAR2	2000	0	0	Free text description of the peer JNDMS.
inputFilter	VARCHAR2	2000	0	0	Definition of any input filters.
outputFilter	VARCHAR2	2000	0	0	Definition of any output filters.

Connections

Connector	Source	Target
Dependency Source -> Destination	Application:JNDMSPeer	Application:JNDMSPeer_ID_seq
Dependency Source -> Destination	Situational Awareness Data Sharing:Replicate data with peer JNDMS	Application:JNDMSPeer

Operations

Method	Parameter
PK_Peer()	NUMBER ID [in]
TRG_PreInsert_JNDMSPeer before insert()	

3.5.4.5.10.14 JNDMSPeer_ID_seq

Type: **Class**

Stereotype: «db_sequence»

Notes:

This is a sequence generator to provide a unique identifier for JNDMS Peer records.

Connections

Connector	Source	Target
<u>Dependency</u> Source -> Destination	Application:JNDMSPeer	Application:JNDMSPeer_ID_seq

3.5.4.5.10.15 JNDMSRule

Type: **Class**

Stereotype: «table»

Notes:

The JNDMS rules will maintain the rules required by the DSS.

Attributes

Name	Type	Length	Precision	Scale	Notes
type	VARCHAR2	3	0	0	Allowable values are DSS and SIM. SIM rules may not need to be stored in the DB?
ruleID	NUMBER	0	10	0	The rule ID is a unique number to be associated with each rule.
created	TIMESTAMP	0	0	0	Timestamp record created.
modified	TIMESTAMP	0	0	0	Timestamp last modified.
rule	VARCHAR2	2000	0	0	Rule data.

Connections

Connector	Source	Target
<u>Dependency</u> Source -> Destination	Presentation Visualization Alerting:Create and Update JNDMS Rules	Application:JNDMSRule

Operations

Method	Parameter
PK_JNDMSRules()	<u>VARCHAR2</u> type [in] <u>NUMBER</u> ruleID [in]
TRG_PreInsert_JNDMSRule before insert()	

3.5.4.5.10.16 StatusHistory

Type: **Class**

Stereotype: «table»

Notes:

This will record the history of the global status and update 1) when a status changes or 2) at a regular interval.

Attributes

Name	Type	Length	Precision	Scale	Notes
Type	VARCHAR2	50			The type of Status being recorded. One of Asset, Operation, OpArea, Incident
InfoDate	TIMESTAMP				The time of the snapshot of the status
LowValue	NUMBER		8	2	The count / value stored that refers to a lower value (green)
MidValue	NUMBER		8	2	The value stored for something Amber
HighValue	NUMBER		8	2	The value stored for something in the Red
Name	VARCHAR2	255			The name of a Location, or Operation that is associated with the details. NULL for global data

Connections

Connector	Source	Target
Dependency Source -> Destination	Application:StatusHistory	Application:StatusHistory_Procedures

Operations

Method	Parameter
TRG_PreInsertStatusHistory before insert()	

3.5.4.5.10.17 StatusHistory_Procedures

Type: **Class**

Stereotype: «stored procedures»

Notes:

Connections

Connector	Source	Target
Dependency Source -> Destination	Application:StatusHistory	Application:StatusHistory_Procedures

Operations

Method	Parameter
StatusHistoryAddAsset()	
StatusHistoryAddIncident()	
StatusHistoryAddOpArea()	
StatusHistoryAddOpAreaIncident()	
StatusHistoryAddOpAreaRisk()	
StatusHistoryAddOperation()	
StatusHistoryAddOperationRisk()	
StatusHistoryAddOpIncident()	
StatusHistoryStatusUpdate()	

3.5.4.5.10.19 Source

Type: **Class**

Stereotype: «table»

Notes:

Attributes

Name	Type	Length	Precision	Scale	Notes
datasource	VARCHAR2	50			A source of data
lastContact	TIMESTAMP				the last time the source contacted us with any query
lastUpdate	TIMESTAMP				the last time any data was submitted
lastID	VARCHAR2	500			the last external ID the source wanted stored for future use

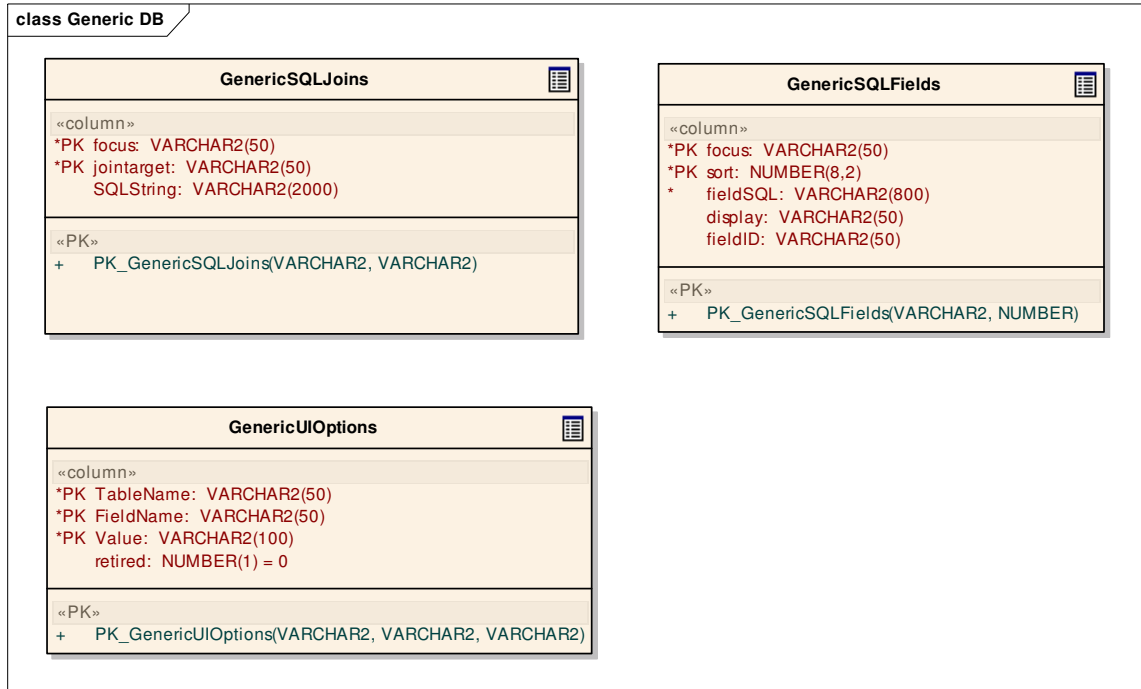
Operations

Method	Parameter
PK_Datasource()	<u>VARCHAR2</u> datasource [in]

3.5.4.5.10.20 Generic DB

Generic DB - (Logical diagram)

This contains table definitions for anything relating to the Generic DB implementation. These tables will be populated to control the focus, columns, and table joins.



3.5.4.5.10.21 GenericSQLFields

Type: **Class**

Stereotype: «table»

Notes:

This details the fields that will be visible when viewing a list of any 'focus' item.

Attributes

Name	Type	Length	Precision	Scale	Notes
focus	VARCHAR2	50			The name of the table that is considered the 'focus' of the view.
sort	NUMBER		8	2	Allows control over the order the field will appear in the ResultSet
fieldSQL	VARCHAR2	800			The SQL used to create the field. This should include the table specifier
display	VARCHAR2	50			The name that should appear as the column header in a list view.
fieldID	VARCHAR2	50			A unique non-spaced name used to identify the field in the ResultSet

Operations

Method	Parameter
PK_GenericSQLFields()	<u>VARCHAR2</u> focus [in]
	<u>NUMBER</u> sort [in]

3.5.4.5.10.22 GenericSQLJoins

Type: **Class**

Stereotype: «table»

Notes:

This will contain one step joins between any two tables. It may not be possible to make this generic for linking in both directions, so we should assume every join between two tables will have two entries in this table.

Attributes

Name	Type	Length	Precision	Scale	Notes
focus	VARCHAR2	50			The table name of the primary focus of a join. This is the 'where are we' portion of a SQL join - 'Where are we, and where do we want to go.'
jointarget	VARCHAR2	50			The table name of another focus that represents the intended destination of a join. 'Where are we, and where are we going'.
SQLString	VARCHAR2	2000			This is the JOIN SQL query used directly in an SQL statement

Operations

Method	Parameter
PK_GenericSQLJoins()	<u>VARCHAR2</u> focus [in] <u>VARCHAR2</u> jointarget [in]

3.5.4.5.10.23 GenericUIOptions

Type: **Class**

Stereotype: «table»

Notes:

Stores additional dropdown fields that need to be shown to the user during manual entry.

Attributes

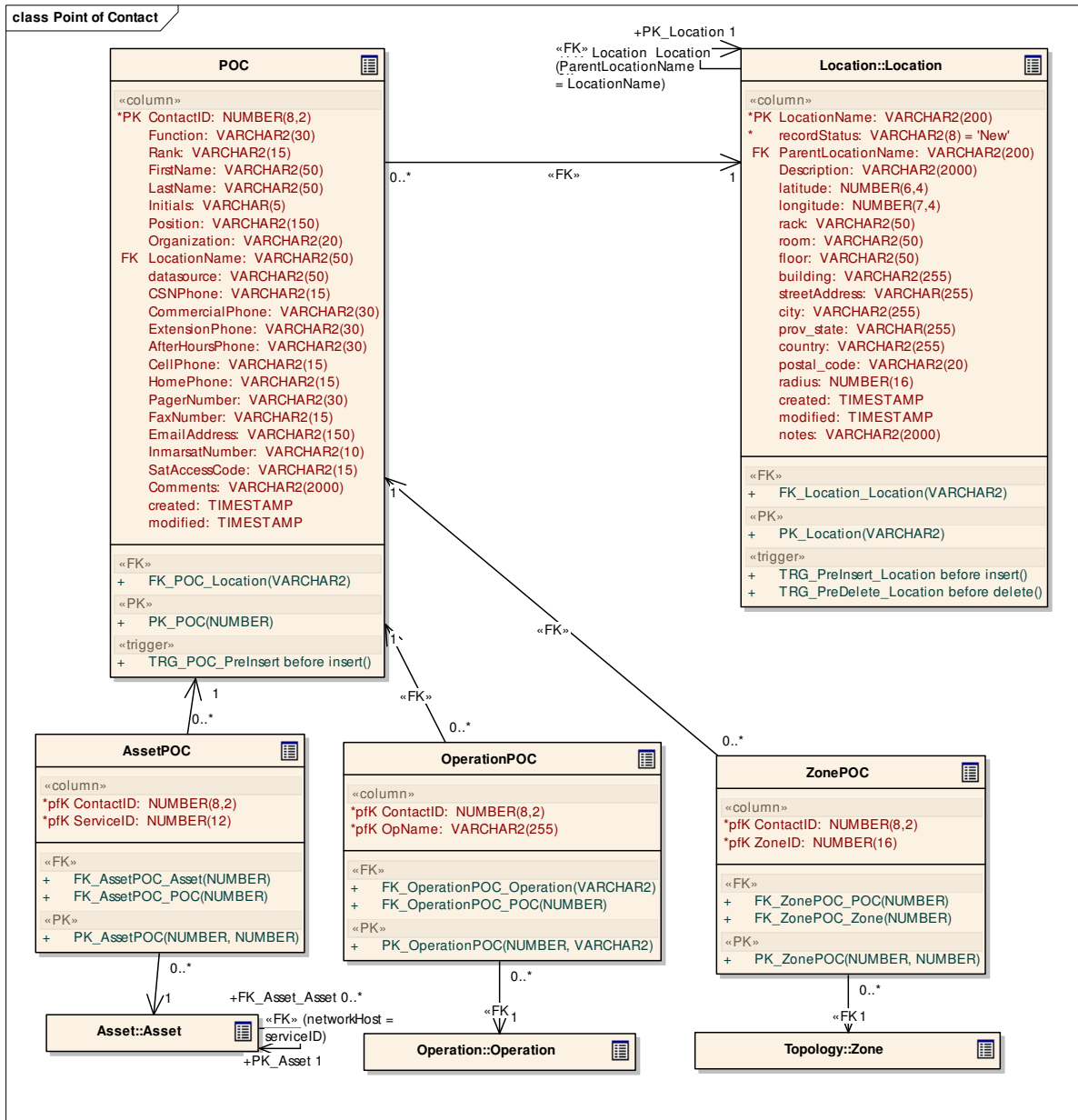
Name	Type	Length	Precision	Scale	Notes
TableName	VARCHAR2	50			The name of the table the data is in
FieldName	VARCHAR2	50			The field the additional data is in
Value	VARCHAR2	100			The additional data to be shown in the dropdown
retired	NUMBER		1	0	0 for active, 1 for retired

Operations

Method	Parameter
PK_GenericUIOptions()	<p><u>VARCHAR2</u> TableName [in]</p> <p><u>VARCHAR2</u> FieldName [in]</p> <p><u>VARCHAR2</u> Value [in]</p>

3.5.4.5.10.24 Point of Contact

Point of Contact - (Logical diagram)



3.5.4.5.10.24.1 AssetPOC

Type: **Class**

Stereotype: «table»

Notes:

Handles a many to many relationship between contacts and assets

Attributes

Name	Type	Length	Precision	Scale	Notes
ContactID	NUMBER		8	2	Foreign key to POC table
ServiceID	NUMBER		12	0	Foreign Key to Asset table

Connections

Connector	Source	Target
<u>Association</u> (ContactID = ContactID) Source -> Destination	Point of Contact:AssetPOC	Point of Contact:POC
<u>Association</u> (ServiceID = serviceID) Source -> Destination	Point of Contact:AssetPOC	Asset:Asset

Operations

Method	Parameter
FK_AssetPOC_Asset()	<u>NUMBER</u> ServiceID [in]
FK_AssetPOC_POC()	<u>NUMBER</u> ContactID [in]
PK_AssetPOC()	<u>NUMBER</u> ContactID [in] <u>NUMBER</u> ServiceID [in]

3.5.4.5.10.24.2 OperationPOC

Type: **Class**

Stereotype: «table»

Notes:

Multiple POCs can be associated with any Operation

Attributes

Name	Type	Length	Precision	Scale	Notes
ContactID	NUMBER		8	2	Foreign key to POC table
OpName	VARCHAR2	255			Foreign key to Operation table

Connections

Connector	Source	Target
<u>Association</u> (OpName = name) Source -> Destination	Point of Contact:OperationPOC	Operation:Operation
<u>Association</u> (ContactID = ContactID) Source -> Destination	Point of Contact:OperationPOC	Point of Contact:POC

Operations

Method	Parameter
FK_OperationPOC_Operation()	<u>VARCHAR2</u> OpName [in]
FK_OperationPOC_POC()	<u>NUMBER</u> ContactID [in]
PK_OperationPOC()	<u>NUMBER</u> ContactID [in] <u>VARCHAR2</u> OpName [in]

3.5.4.5.10.24.3 POC

Type: **Class**

Stereotype: «table»

Notes:

This table contains all the names and contact information of people who are responsible for assets and / or operation details.

Attributes

Name	Type	Length	Precision	Scale	Notes
ContactID	NUMBER		8	2	This is the autonumber ContactID of the record.
Function	VARCHAR2	30			The function of the person in the position. Can be one of: ISSO - Primary,Organization Contact,Primary Contact,Secondary Contact,ISSO - Secondary,International,Secondary/ Night Contact,ComSec Custodian,Taclane Admin - Primary,Taclane Admin - Secondary
Rank	VARCHAR2	15			The rank of the contact. Ranks include: MBdr,Pte,Cpl,MCpl,Sgt,WO,MWO ,CWO,Lt,Capt,Maj,LCol,Col,BGen, MGen,LGen,Gen,Civ,Lt(N),LCdr,C dr,Capt (N),Adm,PO,CPO 2nd Class,CPO 1st Class,PO2,MS,LS,Lt Cdr,Adj,Matc,Cplc,MSgt
FirstName	VARCHAR2	50			The first name of the contact
LastName	VARCHAR2	50			The last name of the contact
Initials	VARCHAR	5			The initials of the contact
Position	VARCHAR2	150			The name of the job position for the contact
Organization	VARCHAR2	20			The name of the organization to which the contact belongs. Values include: CFCS,CFIOG,CFNOC,DDCEI,DN IS 4,DIMTPS,IM

DN0678
Issue 4/2: 06 February 2012

Name	Type	Length	Precision	Scale	Notes
					SEC,NCAS,NSD,76 CommGp,NDCC,J6,CFCS SP 3,DSB LCSF,CFCS Sp 2,NCIS TUNNEYS,764 CommSqn
LocationName	VARCHAR2	50			The Location Name foreign key for the location of the contact.
datasource	VARCHAR2	50			The name of source of data for this contact. Where/whom the information came from.
CSNPhone	VARCHAR2	15			The CSN Phone number of the contact
CommercialPhone	VARCHAR2	30			The commercial phone number of the contact
ExtensionPhone	VARCHAR2	30			The extension number for the phone number
AfterHoursPhone	VARCHAR2	30			The after hours phone number of the contact
CellPhone	VARCHAR2	15			The cellular phone number of the contact
HomePhone	VARCHAR2	15			The home phone number of the contact
PagerNumber	VARCHAR2	30			The pager number of the contact
FaxNumber	VARCHAR2	15			The fax number of the contact
EmailAddress	VARCHAR2	150			The email address of the contact
InmarsatNumber	VARCHAR2	10			The inmarsat number for the contact
SatAccessCode	VARCHAR2	15			The satellite access code for the contact
Comments	VARCHAR2	2000			Any additional comments relating to the contact
created	TIMESTAMP				Date the POC was first entered
modified	TIMESTAMP				Date the POC entry was last modified

Connections

Connector	Source	Target
<u>Association</u> (LocationName = LocationName) Source -> Destination	Point of Contact:POC	Location:Location
<u>Association</u> (ContactID = ContactID) Source -> Destination	Point of Contact:ZonePOC	Point of Contact:POC
<u>Association</u> (ContactID = ContactID) Source -> Destination	Point of Contact:AssetPOC	Point of Contact:POC
<u>Association</u> (ContactID = ContactID) Source -> Destination	Point of Contact:OperationPOC	Point of Contact:POC

Operations

Method	Parameter
FK_POC_Location()	<u>VARCHAR2</u> LocationName [in]
PK_POC()	<u>NUMBER</u> ContactID [in]
TRG_POC_PreInsert before insert()	

3.5.4.5.10.24.4 ZonePOC

Type: **Class**

Stereotype: «table»

Notes:

Multiple POCs can be associated with any Zone

Attributes

Name	Type	Length	Precision	Scale	Notes
ContactID	NUMBER		8	2	Foreign key to POC table
ZoneID	NUMBER		16	0	Foreign key to Zone table

Connections

Connector	Source	Target
<u>Association</u> (ContactID = ContactID) Source -> Destination	Point of Contact:ZonePOC	Point of Contact:POC
<u>Association</u> (ZoneID = zoneID) Source -> Destination	Point of Contact:ZonePOC	Topology:Zone

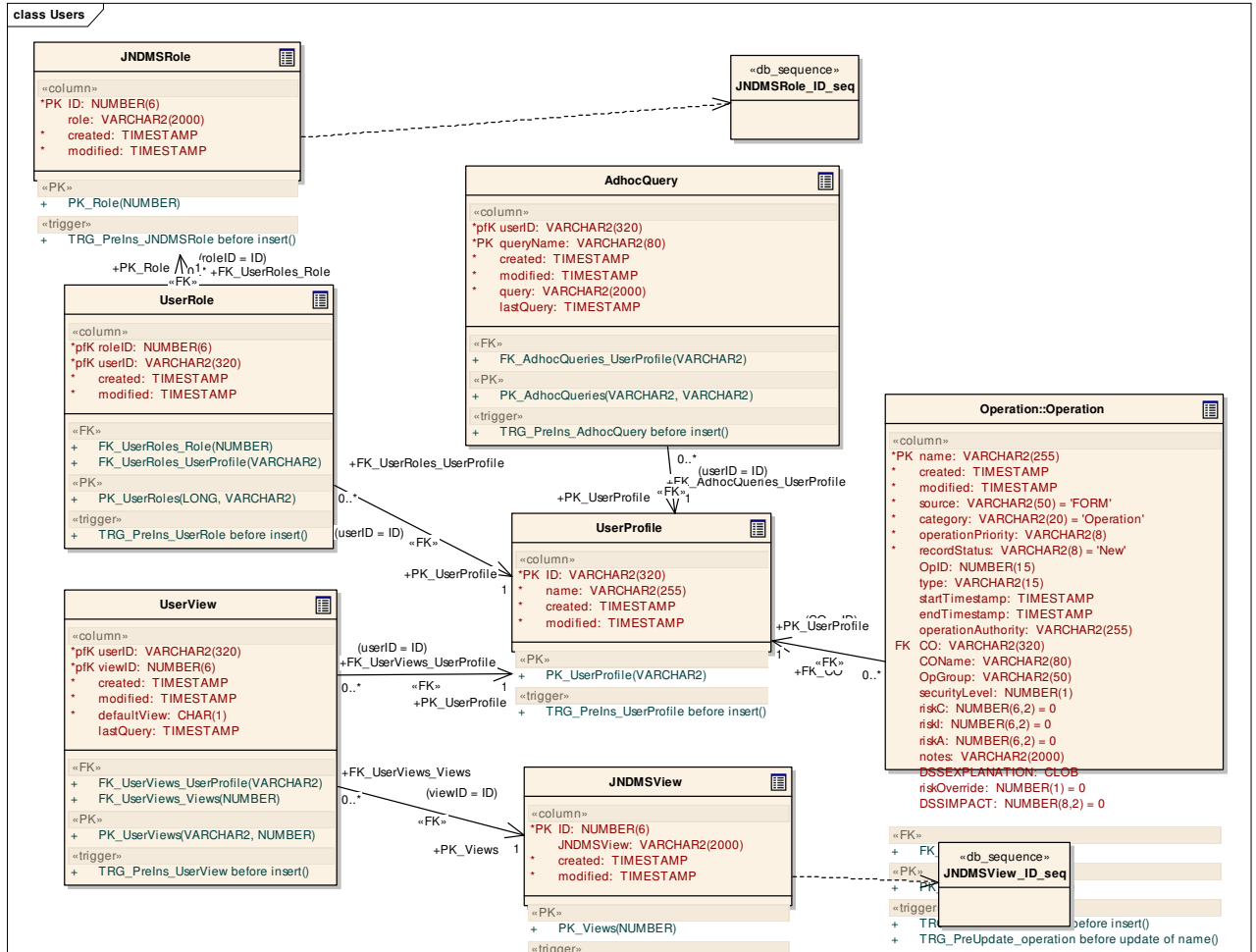
Operations

Method	Parameter
FK_ZonePOC_POC()	<u>NUMBER</u> ContactID [in]
FK_ZonePOC_Zone()	<u>NUMBER</u> ZoneID [in]
PK_ZonePOC()	<u>NUMBER</u> ContactID [in] <u>NUMBER</u> ZoneID [in]

3.5.4.5.10.24.5 Users

The Users package contains those elements to provide application support for management of user configurations.

Users - (Logical diagram)



3.5.4.5.10.24.6 AdhocQuery

Type: **Class**

Stereotype: «table»

Notes:

Ad hoc queries can be defined by the user and stored for later re-query.

Attributes

Name	Type	Length	Precision	Scale	Notes
userID	VARCHAR2	320	0	0	User maintaining the ad hoc query.
queryName	VARCHAR2	80	0	0	Free text user specified unique name for the query.
created	TIMESTAMP	0	0	0	Timestamp of record creation.
modified	TIMESTAMP	0	0	0	Timestamp of last modification (excluding last query).
query	VARCHAR2	2000	0	0	Query definition.
lastQuery	TIMESTAMP	0	0	0	Timestamp from last time the query was run by the user.

Connections

Connector	Source	Target
<u>Association</u> (userID = ID) Source -> Destination	Users:AdhocQuery	Users:UserProfile

Operations

Method	Parameter
FK_AdhocQueries_UserProfile()	<u>VARCHAR2</u> userID [in]
PK_AdhocQueries()	<u>VARCHAR2</u> userID [in] <u>VARCHAR2</u> queryName [in]
TRG_PreIns_AdhocQuery before insert()	

3.5.4.5.10.25 JNDMSRole

Type: **Class**

Stereotype: «table»

Notes:

User roles defined in the database.

Attributes

Name	Type	Length	Precision	Scale	Notes
ID	NUMBER	0	6	0	Role identification.
role	VARCHAR2	2000	0	0	Role definition.
created	TIMESTAMP	0	0	0	Timestamp of record creation.
modified	TIMESTAMP	0	0	0	Timestamp of last modification.

Connections

Connector	Source	Target
<u>Association</u> (roleID = ID) Source -> Destination	Users:UserRole	Users:JNDMSRole
<u>Dependency</u> Source -> Destination	Users:JNDMSRole	Users:JNDMSRole_ID_seq

Operations

Method	Parameter
PK_Role()	<u>NUMBER</u> ID [in]
TRG_PreIns_JNDMSRole before insert()	

3.5.4.5.10.25.1 JNDMSRole_ID_seq

Type: **Class**

Stereotype: «*db_sequence*»

Notes:

Connections

Connector	Source	Target
<u>Dependency</u> Source -> Destination	Users:JNDMSRole	Users:JNDMSRole_ID_seq

3.5.4.5.10.25.2 JNDMSView

Type: **Class**

Stereotype: «table»

Notes:

JNDMS views.

Attributes

Name	Type	Length	Precision	Scale	Notes
ID	NUMBER	0	6	0	View identifier.
JNDMSView	VARCHAR2	2000	0	0	View definition.
created	TIMESTAMP	0	0	0	Timestamp of record creation.
modified	TIMESTAMP	0	0	0	Timestamp of last modification.

Connections

Connector	Source	Target
<u>Dependency</u> Source -> Destination	Users:JNDMSView	Users:JNDMSView_ID_seq
<u>Association</u> (viewID = ID) Source -> Destination	Users:UserView	Users:JNDMSView

Operations

Method	Parameter
PK_Views()	<u>NUMBER</u> ID [in]
TRG_PreIns_JNDMSView before insert()	

3.5.4.5.10.26 JNDMSView_ID_seq

Type: **Class**

Stereotype: «db_sequence»

Notes:

Connections

Connector	Source	Target
<u>Dependency</u> Source -> Destination	Users:JNDMSView	Users:JNDMSView_ID_seq

3.5.4.5.10.27 UserProfile

Type: **Class**

Stereotype: «table»

Notes:

A user profile represents a specific JNDMS user.

Attributes

Name	Type	Length	Precision	Scale	Notes
ID	VARCHAR2	320	0	0	User ID (e-mail address).
name	VARCHAR2	255	0	0	Users real name.
created	TIMESTAMP	0	0	0	Timestamp of last modification.
modified	TIMESTAMP	0	0	0	Timestamp of last modification.

Connections

Connector	Source	Target
<u>Association</u> (CO = ID) Source -> Destination	Operation:Operation	Users:UserProfile
<u>Association</u> (userID = ID) Source -> Destination	Users:UserView	Users:UserProfile
<u>Association</u> (userID = ID) Source -> Destination	Users:UserRole	Users:UserProfile
<u>Association</u> (userID = ID) Source -> Destination	Users:AdhocQuery	Users:UserProfile

Operations

Method	Parameter
PK_UserProfile()	<u>VARCHAR2</u> ID [in]
TRG_PreIns_UserProfile before insert()	

3.5.4.5.10.28 UserRole

Type: **Class**

Stereotype: «table»

Notes:

Mapping of a user to 1 or more roles and a role to one or more users.

Attributes

Name	Type	Length	Precision	Scale	Notes
roleID	NUMBER	0	6	0	Role identified.
userID	VARCHAR2	320	0	0	Role definition.
created	TIMESTAMP	0	0	0	Timestamp of record creation.
modified	TIMESTAMP	0	0	0	Timestamp of last modification.

Connections

Connector	Source	Target
<u>Association</u> (userID = ID) Source -> Destination	Users:UserRole	Users:UserProfile
<u>Association</u> (roleID = ID) Source -> Destination	Users:UserRole	Users:JNDMSRole

Operations

Method	Parameter
FK_UserRoles_Role()	<u>NUMBER</u> roleID [in]
FK_UserRoles_UserProfile()	<u>VARCHAR2</u> userID [in]
PK_UserRoles()	<u>LONG</u> roleID [in] <u>VARCHAR2</u> userID [in]
TRG_PreIns_UserRole before insert()	

3.5.4.5.10.29 UserView

Type: **Class**

Stereotype: «table»

Notes:

Mapping of users to views and views to users.

Attributes

Name	Type	Length	Precision	Scale	Notes
userID	VARCHAR2	320	0	0	User ID (e-mail address).
viewID	NUMBER	0	6	0	View Identification.
created	TIMESTAMP	0	0	0	Timestamp of record creation
modified	TIMESTAMP	0	0	0	Timestamp of last modification (excluding last query).
defaultView	CHAR	1	0	0	Is this the users default view? Y or N.
lastQuery	TIMESTAMP	0	0	0	This records the last time the view was queried to determine if there have been any changes.

Connections

Connector	Source	Target
<u>Association</u> (userID = ID) Source -> Destination	Users:UserView	Users:UserProfile
<u>Association</u> (viewID = ID) Source -> Destination	Users:UserView	Users:JNDMSView

Operations

Method	Parameter
FK_UserViews_UserProfile()	<u>VARCHAR2</u> userID [in]
FK_UserViews_Views()	<u>NUMBER</u> viewID [in]
PK_UserViews()	<u>VARCHAR2</u> userID [in] <u>NUMBER</u> viewID [in]
TRG_PreIns_UserView before insert()	

This is a Java component making use of the J2EE middleware stack and the support within the Java language for web services using the Java Web Services Developer Pack (WSDP). The external interfaces for the JNDMS Services are as follows:

- Web Services using SOAP for inputs and queries
- JDBC to interface with the Data Warehouse

JDBC is the standard API used to perform database operations from a Java environment. JDBC is supported by both J2EE and J2SE standards and enables the “Write Once, Run Anywhere” capabilities to access enterprise data. For additional information on the JDBC API, see < <http://java.sun.com/products/jdbc/overview.html>>.

3.6.2 Interfaces

The JNDMS Services will export SOAP services that define the inputs and queries required. Section 3.6.2.2 identifies the methods defined in the SOAP interface and the information flow through each one.

The interface defining what operations are available and the structure of each request and response is defined using the Web Service Definition Language. The JSS exports two core interfaces, the JSS and the JSS Queue interfaces. The contents of the interface can be found in Annex F.

3.6.2.1 JSS Queue Interface Summary

The JSS Queue Interface provides several methods to query and manage the JSS internal queues (see Section 3.6.3 for internals of the JSS and managed queues). Any event that has the potential of writing to the data model or being shared with other systems will be managed by the event queues. Any event that is managed by the event queues will have a unique identifier that will be returned when the event is submitted.

The following methods are available:

- View(event ids). This will return the details of the requested events. The event ID input can be either a comma separated list of events or ‘all’. The returned information includes the source of the event, the ID, its description, its processing stage (see section 3.6.3), its current result (pending, error, done, etc) and the result of any sharing operation.
- getStatus (event ids). This returns a summarized version of the information available from the view method.
- Remove(event id). This method allows events to be removed from the processing queues.
- Submit(event details). This interface was initially used to implement the sharing component, however it has been deprecated in favour of using the JSS interface directly.

3.6.2.2 JSS Interface Summary

The JSS interface provides the methods to submit data to JNDMS and to interact with the system. See the entries in Table 16: JSS Data Inputs for the definitions of the inputs available.

The following operations are available²:

- SIMAlarm: Submit security related events.
- EIMEvent: Submit infrastructure related events.
- CAPEvent: Submit generalized events.
- VulnerabilityDefintion: Submit either new or updated vulnerability definitions.
- VulnerabilityScan: Submit vulnerability scans.
- MalwareDefinition: Submit either new or updated malware definitions.
- EIMXMLEvent: This provides a very generic interface that allows fine grained updates to the JNDMS data model. The system defines an XML schema that allows queries and data updates to be performed.
- OperationReport: This allows the submission of operations data.
- SafeguardDefinition: This allows the submission of safeguard data, including firewall details.
- AssetReport: This provides the ability to submit software asset inventory updates. This would include any changes to the software inventory on a system.
- DefensivePosture: This provides some minor configuration or control of the defensive posture analysis. Currently the only options are to force a recalculation of the analysis results or to enable/disable the analysis.

² Based on jss.wsdl

The data inputs as shown in Table 16 will be supported as defined by the JSS WSDL (see Annex F):

Table 16: JSS Data Inputs

Information	Method	Associated Schema	Notes
IT Infrastructure Availability	EIMEvent	JSS WSDL	IT Infrastructure availability includes status of servers, workstations and services. The availability may be identified specifically as available or not available, or it may be identified as unknown.
IT Infrastructure Discovery	EIMEvent, EIMXMLEvent	JSS WSDL, JNDMS XML	<p>The discovery of new assets can be reported to JNDMS in two different methods. The first method is through the EIMEvent method in which individual events are reported as they occur. This is the primary interface to the Spectrum and NSM discovery tools.</p> <p>The other method is through the XML reports defined in the EIMXML schema. This allows larger single reports and allows for assets to be reported in conjunction with other reports, such as operation definitions.</p>
IT Infrastructure Performance Alert	EIMEvent	JSS WSDL	This allows individual performance alerts to be reported.

Information	Method	Associated Schema	Notes
IT Infrastructure Software Asset Information	AssetReport, EIMXMLEvent	JSS WSDL, JNDMS XML, JNDMS Asset XML	<p>The details of the software installed on servers, workstations or other equipment is generally reported via the asset report.</p> <p>Software, like other assets, can also be reported via the EIMXML schema. This allows particular software and services to be defined along with operation reports, for example.</p>
Operation Definition	OperationReport, EIMXMLEvent	JSS WSDL, JNDMS XML	<p>The operation report currently defines an operation. This includes the detailed information of the operation, its units, planned events, overall priority and its relationship to assets and services.</p> <p>The relationships defined to the assets, including software and services; include how the importance each of these assets is to the execution of this operation.</p>
Vulnerability Data	VulnerabilityDefinition	JSS WSDL, NVD/CVE XML	This is the method that XML reports of vulnerability definitions, such as the CVE, are reported.
Vulnerability Instance Data (scans)	VulnerabilityScan	JSS WSDL, JNDMS Vulnerability Scans	<p>This is the method that XML reports of vulnerability scans are reported to JNDMS.</p> <p>JNDMS defines its own vulnerability scan XML with a translator to ingest Nessus vulnerability scans.</p>

Information	Method	Associated Schema	Notes
Safeguard Data	SafeguardDefinition	JSS WSDL, Non XML input	The safeguard report can be used to define zones (and zone related information such as zone rules) as well as particular safeguards and their impact on known vulnerabilities.
Security events and alarms	SIMAlarm	JSS WSDL	Security events and alarms from the Security Information Management subsystem are reported through this interface.
Malware Definition	MalwareDefinition	JSS WSDL, CME XML	Malware definitions, through the CME, are reported through this interface.
Request to update risk analysis	DefensivePosture	JSS WSDL	This method is currently used to ensure that all impacts have been analyzed by the DSS. Its use should be considered temporary and should not be used in normal operation.
Generic alerts (Common Alerting Protocol)	CAPEvent	JSS WSDL, CAP XML	This allows generic events, especially those with geographic components, to be reported. The schema currently used is defined by the Common Alerting Protocol. This is partially complete is cycle 3.
Location Information	EIMXMLEvent	JSS WSDL, JNDMS XML	Generic location information, such as the lat/lon of a city, can be reported through this interface. The particular location of assets is done as part of the operation report.

Information	Method	Associated Schema	Notes
Services	EIMXMLEvent	JSS WSDL, JNDMS XML	Services are reported through this interface.
Service relationships (dependencies and redundancies)	EIMXMLEvent	JSS WSDL, JNDMS XML	The relationships between services, including dependent and redundant relationships are reported through this interface.
Relationship editing	EIMXMLEvent	JSS WSDL, JNDMS XML	<p>This interface allows relationships between JNDMS entities to be manipulated. For example adding assets to an operation would be considered editing relationships.</p> <p>The initial relationships include assets to an operation and locations to an operation; however the interface defined is generic and would not change as more relationships are added.</p>

3.6.2.3 JSS XML Schemas

There are a number of XML schemas used by the JSS operations to define complex inputs. This section will outline the schemas in use. Refer to Table 16 to identify which schemas are required for the JSS operations. The schema files for JNDMS custom interfaces are part of the web application, stored in WebContent/WEB-INF/xml.

Schema	Reference	Notes
JSS WSDL	Section 3.6.2.2, Section 3.6.2.3.1	This is the WSDL for the JSS Web Services. This is defined by JNDMS
JNDMS XML	Section 3.6.2.3.1 jndms.xsd	This is a custom JNDMS schema used to define many JNDMS relationships.
JNDMS Assets XML	Section 3.6.2.3.2 ca_assets.xsd	This is a custom JNDMS schema used to define software asset inventories.
JNDMS Vulnerability Scan XML	Section 3.6.2.3.3 jndmsvulnscn.xsd	This is a custom JNDMS schema used to define vulnerability scans.
NVD/CVE XML	http://nvd.nist.gov/schema/nvdcve.xsd	This is an industry standard schema.
CAP XML	http://www.incident.com/cap/docs.html	This is an industry standard schema.
CME XML	http://cme.mitre.org/	This is an industry standard schema.

3.6.2.3.1 JNDMS XML

The JNDMS XML format is designed to be able to represent a number of the common JNDMS entities and relationships. At the root of the document is the 'jndms' type that can contain the following sections:

- **Locations:** This can define locations as required by JNDMS including parent entries, names and geographic locations. These locations will be entered into the JNDMS data warehouse if they don't already exist. Any locations referenced in other sections should be defined here.
- **Assets:** This section defines asset references that are to be used in other sections. This section can create new assets or can use wild cards to reference existing assets. For example all assets of a given type on a specified subnet could be given a name. This name would be used by other sections in this XML and when it is processed all matching assets would be processed.
- **Services:** This allows services, including their dependencies and redundancies to be defined.
- **Operations:** This allows all attributes required by JNDMS to define an operation to be specified.
- **Relationships:** This is a very generic and low level, XML query. Each of these defines edits to relationships within the JNDMS data model. Relationships are generally defined as links to intersection tables in the data model. This operation is often used by the portal to submit user edits.
- **Attributes:** This is another very generic XML query. This allows attributes of common JNDMS entities to be changed. This operation is often used by the portal to submit user edits.

Example defining services

The following is an annotated example of defining services using the JNDMS XML schema:

Provide XML header information that references the schema

```
<?xml version="1.0"?>  
<jndms xmlns="http://jndms"  
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
  xsi:schemaLocation="http://jndms jndms.xsd">
```

The 'asset' section provides the ability to either create new assets (hardware or software) or to reference existing assets within the database. The id given to assets or asset groups is used in further section to create services.

```
<assets>
```


Existing assets are created first. These assets (referenced using the 'assetRef' tag) can represent single assets or groups of assets. These assets are searched using the provided criteria.

```
<existingAssets>
  <assetRef>
    <id>Deployed Portal Workstations</id>
    <ip_range>10.136.50.100 - 10.136.50.200</ip_range>
    <ip_range>10.136.51.100 - 10.136.51.200</ip_range>
  </assetRef>
  <assetRef>
    <id>Deployed Email Workstations</id>
    <ip_range>10.136.54.100 - 10.136.54.200</ip_range>
  </assetRef>
  <assetRef>
    <id>Exchange1</id>
    <net><ip_address>192.168.200.20</ip_address></net>
  </assetRef>
  <assetRef>
    <id>Exchange2</id>
    <net><ip_address>192.168.201.20</ip_address></net>
  </assetRef>
  <assetRef>
    <id>DNS1</id>
    <net><ip_address>10.32.12.26</ip_address></net>
  </assetRef>
  <assetRef>
    <id>DNS2</id>
    <net><ip_address>10.136.30.22</ip_address></net>
  </assetRef>
</existingAssets>
```

New assets can also be created. New assets can be physical assets such as hardware or software and they can also be purely logical assets such as services.

```
<asset>
  <method>service</method>
  <id>Exchange</id>
  <name>Exchange</name>
  <category>
    <type>network service</type>
    <category>service</category>
  </category>
</asset>
```

Services are defined next. This section will provide the details of how services related to one another. All dependant and redundant links are defined in this section.

```
<services>
```

The service definition in this example is for DNS. This defines a service that references the assets above and, in this case, creates a single logical service that has two redundant servers.

```
<service>
  <provider>
    <method>service</method>
    <id>DNS</id>
    <name>DNS</name>
  </provider>
  <link type="redundant">
    <source>
      <id>DNS</id>
    </source>
    <destination>
      <id>DNS1</id>
    </destination>
    <destination>
      <id>DNS2</id>
    </destination>
  </link>
</service>
```

The next example shows the creation of an 'Exchange' service. This is a simple definition for the example, however it could reference the assets to identify the physical assets that it depends upon, including any redundancies.

```
<service>
  <provider>
    <method>service</method>
    <id>Exchange</id>
    <name>Exchange</name>
  </provider>
</service>
```

DN0678
Issue 4/2: 06 February 2012

The next step is to provide the service that defines the relationship to the clients. In this case we will define a generic 'DND email' service that groups of workstations depend upon, and it, in turn, depends on the created exchange service. This combination completes the link from the client workstation to all servers required.

```
<service>
  <provider>
    <method>service</method>
    <id>DND Email</id>
    <name>DND Email</name>
  </provider>
  <link>
    <source>
      <id>Deployed Email Workstations</id>
    </source>
    <destination>
      <id>DND Email</id>
    </destination>
  </link>
  <link>
    <source>
      <id>DND Email</id>
    </source>
    <destination>
      <id>Exchange</id>
    </destination>
  </link>
</service>
</services>
</jndms>
```

Example of operation definition

The following is an annotated example of using the JNDMS schema to define an operation.

XML header information.

```
<?xml version="1.0"?>
<jndms
xmlns="http://jndms"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://jndms jndms.xsd">
```

```
<operations>
```

We will define an operation 'CFIOG', that initially provides key attributes.

```
<operation>
  <name>CFIOG</name>
  <source> J6 </source>
  <category>operation</category>
  <priority>medium</priority>
  <type>domestic</type>
  <start_date>2006-01-02T00:00:00</start_date>
  <end_date>2010-12-31T23:59:00</end_date>
  <authority>J6</authority>
  <co>
    <full_name> Col John Doe </full_name>
  </co>
  <op_group>CFIOG</op_group>
  <poc>
    <first_name> J. </first_name>
    <last_name>Bloggins</last_name>
  </poc>
  <risk_profile><c>1.0</c><i>1.0</i><a>1.0</a></risk_profile>
  <notes> Canadian Forces Information Operations Group </notes>
```

The 'event' section defines the timing of the operation or events within the operation.

```
<event>
  <id>1</id>
  <description>CFNOC assigned to monitor CND</description>
  <start_time>2007-06-12T00:00:00</start_time>
  <end_time>2010-12-31T23:59:00</end_time>
</event>
```

DN0678
Issue 4/2: 06 February 2012

The 'unit' section will define units or groups within the operation.

```
<unit>
  <name>CFNOC</name>
  <poc>
    <first_name>J.</first_name>
    <last_name>Bloggins</last_name>
  </poc>
  <notes></notes>
</unit>
<unit>
  <name>Contracted Services</name>
  <poc>
    <first_name>Un</first_name>
    <last_name>Known</last_name>
  </poc>
  <notes>Canadian Forces Network Operation Centre</notes>
</unit>
```

The 'plan' section defines how the events (temporal), the units (who), the location (where) and the assets (what) are combined. The operational plan defines one or more records that must combine the events, units, locations, assets and their importance.

```
<plan>
  <record>
    <event>1</event>
    <unit>CFNOC</unit>
    <location>CFS Lietrim, ON, CA</location>
```

DN0678
Issue 4/2: 06 February 2012

The relationship to assets is defined by either 'capability' or 'provision' entries. The information in each will define the impact of not having the required capability, service, asset, etc. The 'provision' entries, however, also signify that this operation has nominal responsibility for the referenced assets, not just a dependence on them.

```
<capability>
  <id>Marpac C2</id>
  <lossOfLife>>false</lossOfLife>
  <impact>
    <c>low</c>
    <i>low</i>
    <a>low</a>
  </impact>
  <importance>essential</importance>
</capability>
<provision>
  <subnet>192.168.200.0/24</subnet>
  <lossOfLife>>false</lossOfLife>
  <impact>
    <c>low</c>
    <i>low</i>
    <a>low</a>
  </impact>
  <importance>useful</importance>
</provision>
</record>
</plan>
<primary_location>CFS Lietrim, ON, CA</primary_location>
</operation>
</operations>
</jndms>
```

3.6.2.3.2 JNDMS Asset XML

This XML schema can be used to define software asset inventories to JNDMS. This contains information about a scan that may list software found on one or more hosts. Software assets can also be created using the JNDMS XML schema by using the 'assets' section (see previous section).

The following is a small annotated example of an asset inventory using the JNDMS XML format:

XML Header information:

```
<?xml version="1.0"?>
<jndms
xmlns="http://jndms"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://jndms jndms.xsd">
```

This example shows entering (and possibly creating) a software product on a host.

```
<assets>
  <asset>
    <method>software</method>
    <name> Test Software, TMPDATA </name>
    <host>
      <net> <ip_address>1.2.3.4</ip_address> </net>
    </host>
```

The 'create' flag must be set to 'true' for the products to be created.

```
    <product create="true">
      <name> Test Software, TMPDATA </name>
      <vendor> Test Vendor, TMPDATA </vendor>
      <version> 1.2.3 TMPDATA </version>
    </product>
  </asset>
</assets>
</jndms>
```

The following is an example using CA Asset reporting:

```
<?xml version="1.0" encoding="ISO-8859-1"?>

<RESULT>

  <REPORTTEMPLATE>JNDMS - S/W Inventory Report</REPORTTEMPLATE>

  <RESULTNAME>Demo software load</RESULTNAME>

  <RESULTTIME>7/15/2008 12:00:55 AM</RESULTTIME>

  <EXPORTTIME>7/15/2008 12:01:17 AM</EXPORTTIME>

  <ROWCOUNT>2</ROWCOUNT>

  <COLUMNCOUNT>10</COLUMNCOUNT>
```

Each 'RESULTROW' identifies the workstation and software combination.

```
<RESULTROW>
  <Name>DemoWorkstation</Name>
  <Signature_Scan_AllInstalled>Yes</Signature_Scan_AllInstalled>
  <Signature_Scan_AllTitle>Microsoft Windows XP Professional SP2 x86 32
EN</Signature_Scan_AllTitle>
  <Signature_Scan_AllManufacturer>Microsoft Corporation</Signature_Scan_AllManufacturer>
  <Signature_Scan_AllPath>C:\WINDOWS\system32\ntoskrnl.exe</Signature_Scan_AllPath>
  <Signature_Scan_AllDetected_date>11/15/2006 2:02:34 AM</Signature_Scan_AllDetected_date>
  <Signature_Scan_AllSerial></Signature_Scan_AllSerial>
  <Signature_Scan_AllDetected_version>5.1.2600.2180</Signature_Scan_AllDetected_version>
  <Signature_Scan_AllVersion_Label>XP Professional SP2</Signature_Scan_AllVersion_Label>
  <Signature_Scan_AllSoftware_Type>Release</Signature_Scan_AllSoftware_Type>
</RESULTROW>
</RESULT>
```

3.6.2.3.3 JNDMS Vulnerability Scan XML

This XML schema can define the results of a vulnerability scan. It will generally contain information on who performed the scan (scan host), a list of referenced vulnerabilities and a list of hosts (target) that identify vulnerabilities on each.

Scans performed by Nessus can be submitted to the JSS Vulnerability Scan operation, however an XSLT is used to first transform the Nessus input type into this JNDMS custom vulnerability scan schema.

3.6.3 JSS Internals

The primary purpose of the JSS is to manage the event flow and to provide initial support for maintaining the JNDMS data model. To accomplish this task several threads of execution are used which manage a central set of queues. Figure 3-5 shows the relationships between the internal threads and the event queues.

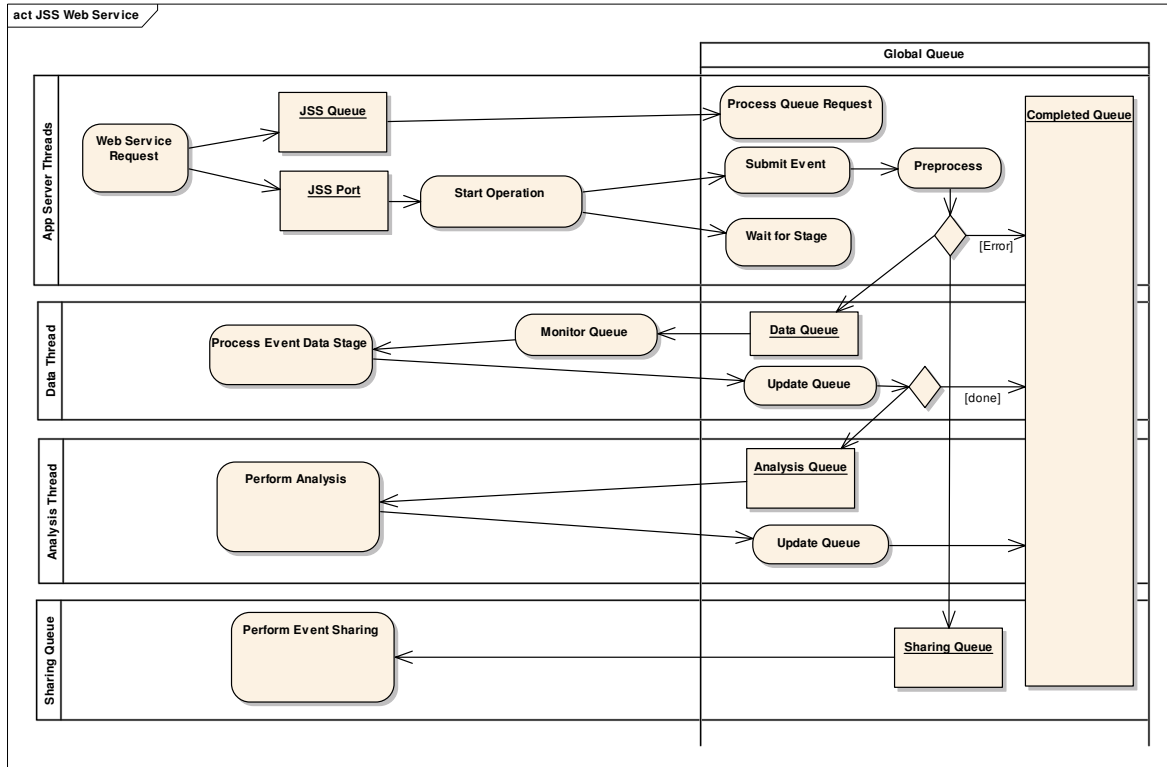


Figure 3-5: JSS threads and queues

The following classes are implemented within JSS/Core:

- EventAnalysisThread.java
- EventDataThread.java
- EventShareThread.java
- JSSEventEntry.java
- JSSEventListener.java
- JSSContextListener.java
- GlobalEventQueue.java (extends JSSEventQueue.java)

Each event that is submitted to the 'JSS Port' (primary JSS interface) that cannot be immediately fulfilled, including the possible requirement to share the event, will be submitted to the global queue and assigned a unique identifier. This identifier will be returned as a result of the web service request and can be used to check on the status of the event at any point.

Each event has a common 'header' portion that consists of the following:

- **Description:** A text description of the event. This provides context information when viewing the event queues.
- **stageWait:** When an event is submitted to the JSS the calling application can request that it doesn't want to wait (`stageNone`) for any event processing or that it doesn't want the called method to return until a specified stage (see section 3.6.3.1 on stages) has completed. This allows the calling application the ability to ensure that the specified stage has completed before the results are returned. This may cause the application to wait (blocking) until the event has been processed, but full results may be available. If results are returned before the event has been completed then partial results will be available, the status will be pending and the event id will be provided.
- **sourceID:** Each source of information to JNDMS must have a unique identifier.
- **sourceAddress:** The IP address of the source of the input information.
- **gatewayID:** This field is used by JNDMS internally to manage peer sharing. This field will identify the forwarding JNDMS system and can be used to prevent loops in event data flows. Users do not have to fill in this field.
- **eventID:** This is the assigned ID for the event. Users will not fill in this field.
- **targetAddress:** This is also used by the JNDMS peer sharing module to identify where this event is destined for.

3.6.3.1 Event stages

Each event in JNDMS is managed through discrete stages. Before any event is managed a pre-processing stage is performed. This stage is responsible for validating the inputs and possibly updating fields where required. If an error occurs the event will not be submitted to the queues. This pre-processing is always done before the initial results are returned to the caller.

The core stages for an event are as follows:

- **Data:** The event is waiting or in the process of data management. In this stage the initial updates to the data model are generally done. This may include just queries, updates or more complex interactions. No analysis or results of the input of data is done at this stage.
- **Analysis.** The event is waiting or in the process of analysis, including correlation, defensive posture and risk analysis. This stage currently attempts to batch any available requests so that the analysis can be done on multiple events at once. For example if one hundred events are waiting when the analysis starts the impact of all events will be considered at once. This is necessary to help scale to larger numbers of events. The analysis is the primary domain of the DSS (see section 3.7).
- **Complete:** Any event that is done processing, either because an error has occurred or it cannot be processed further or because it has been completely analyzed will be marked as complete.

In addition to the core stages that an event can go through there is also a 'sharing' stage. This is done in parallel to the other stages after the pre-processing has validated and possible updated the input event.

3.6.3.2 Event Threads

An event, when it first enters the system, is managed by the application server threads. These threads are created and shared with the primary purpose of ensuring a responsive interface. There can be many of these threads.

After an event has been validated it will enter the JSS global queue. At this point there will be three threads that may work on the event. These threads process the data stage, the analysis stage and the event sharing respectively (see Figure 3-5).

3.7 Decision Support System (DSS)

3.7.1 Overview

The Decision Support System (DSS) component within JNDMS is responsible for analyzing each event sent to JNDMS with the purpose to support the Situational Awareness of the network. The DSS examines events from each of the broad CND domains (operations, infrastructure, vulnerability, safeguards and security events) and provides and updates Situational Awareness through indicators such as risk and impact.

The DSS itself is comprised of a custom Java interface that integrates into the JSS. This component keeps track of the current state of the system so that it can quickly determine the impact or results of a given event. The DSS also has the option to make use of a Business Rules Engine (BRE). The BRE was used as a research tool to examine the effectiveness of rules for the analysis that JNDMS would have to accomplish. During cycle 3 the use of in memory models and of managed graphs (see In memory graphs using JGraphT in Section 3.7.2.3) have proven particularly efficient and flexible to the point where all primary analysis can be done without the rules engine and a full JNDMS can be deployed without this component.

The rule engine, however, remains in JNDMS to allow for future experimentation, especially as data sets grow. The use of the Predictive Analysis Server (see Section 3.7.2.2) would require significant amounts of realistic data over a period of time to evaluate. This was not possible during cycle 3 but the BRE component remains in case this type of analysis could be performed in the future.

3.7.2 COTS Selection

3.7.2.1 CA Cleverpath AION Business Rules Engine

The chosen product for the DSS is CA CleverPath Aion Business Rules Expert (BRE). This CA product allows for all kinds of business logic to be defined and used within any computer driven decision process. A high level summary is provided here³:

"CleverPath Aion BRE is a proven industry leading solution for creating intelligent, rule-based eBusiness applications that capture corporate knowledge and human expertise to drive sound decision-making. CleverPath Aion BRE enables organizations to capitalize on market opportunities, re-engineer business processes and implement new processes with unprecedented speed through component-based development, easy-to-maintain decision tables, dynamic rules, and a sophisticated inference engine that manages and interprets large rule sets. Aion BRE relieves the pressures on already overburdened corporate IT departments by allowing non-technical business personnel to build and maintain their own business rules. As a result, business expertise is married to the

³ <https://solutions.oracle.com/scwar/sc/Solution/SCSP-KTGJNVH.html>

technology, uniting the strengths of business units and IT to create applications that encapsulate the knowledge of your best people.

CleverPath Aion BRE is specifically designed to integrate with other CA products, such as CleverPath Predictive Analysis Server, that can predict and monitor behaviour through adaptive pattern recognition, as well as generating rules from data that can be readily accepted by Aion BRE. This integration further strengthens the ability of Aion BRE in automating complex business processes."

Within the scope of cycle 3 the full possibilities of the BRE and the associated PAS could not be explored and much of the functionality was duplicated using in memory graphs (see section 3.7.2.3).

3.7.2.2 CA CleverPath Predictive Analysis Server (PAS)

Another CA product that could be advantageously used in cooperation with Aion BRE is the CleverPath Predictive Analysis Server, also known as CA Neugents.

In an open-ended environment such as the JNDMS, where large volumes of data have to be examined and correlated, not all of the correlation rules are known at the start. These rules would have to be discovered by analysis of the data. This is the type of application for which Neugents (Cleverpath PAS) is suited.

Once an event has been identified, the Aion rules can perform the severity assessment. In a full-cycle integration of the technologies, Neugents can help to refine the rules used by an Aion knowledge base or even discover new rules.

CleverPath Aion BRE provides two levels of integration with PAS (Neugents):

1. Aion programmers can utilize CA's Application Integration Server (AIS) infrastructure technology through AISL1Lib to access Neugents. This allows an Aion application to include a Neugents Component.
2. A DecisionNeugent Import Wizard (accessible from the Aion IDE Tools menu) for converting a decision tree of a DecisionNeugent into Aion decision tables. This allows Neugents to discover business rules from the data, which are subsequently deployed as Aion rules.

To capture "intelligence", Neugents use pattern recognition and neural network algorithms to identify statistical patterns in the data. This approach differs from the CleverPath Aion BRE approach, which uses a fixed set of conditions that have to be explicitly specified in advance. This fundamental difference in approach gives rise to the following two heuristics regarding the types of applications for which Neugents are best suited and for which CleverPath Aion BRE is best suited.

- Use Neugents when the rules are unknown but can be derived from patterns in data; use Aion when the rules are known.
- Use Aion when the rule changes are precisely defined and must be effected a definitive point in time; use Neugents when the need to change rules is more vaguely defined and the exact nature of the rule change may not be well defined.

In spite of these differences in how Aion BRE and Neugents are used, many applications can make use of both technologies in an integrated fashion. For example:

- **Monitoring.** In a monitoring example, Neugents could be used to predict system performance in the future. Aion could then make decisions about actions to take based on predefined policies and expert advice.
- **Diagnosis.** Aion could be used to diagnose a problem with a piece of equipment and make a recommendation. Neugents could be used to predict the impact on the system of taking that action.

Integration of Aion and Neugents requires the AIS infrastructure, which is only available on supported Windows and Solaris platforms.

The support for the PAS has been part of the design; however the cycle 3 version of JNDMS does not use this component.

3.7.2.3 In memory graphs using JGraphT and JEP

The DSS must know, at any given time, the current state of the network and be able to quickly assess changes or events. Previous to cycle 3 the DSS had concentrated on the building of rules within the BRE, however it became apparent that some types of analysis were better suited to graph processing.

To this end a tool called JGraphT was used. This tool allows the building of directed or undirected graphs and provides a number of common algorithms to process these graphs. In our implementation the network topology is stored as a graph, the service dependency model is stored as a graph and the two views are merged together with the help of the Java math Expression Parser (JEP). This combination allows complex traversal of the graphs.

The JGraphT also provides the ability to clone the graphs in a very efficient manner. This ability allows temporary changes to be made without modifying the original graph. This has proven useful in providing a “what if” type of scenario.

3.7.3 Subsystem Interfaces

The primary subsystem interface for the DSS rules engine is through the JNDMS Services and the Data Warehouse.

3.7.3.1 Data Warehouse Subsystem Interface

The DSS consumes the data collected in the Data Warehouse to make decisions about incident identification, severity assessment and the overall defensive posture. The results of the analysis done by the DSS are then written back to the data warehouse.

3.7.3.2 Options for Invoking Aion BRE as a Component of the DSS

The Rules-Base component of the DSS needs to provide the following capability: process each incident identified by the EIM, SIM and Trouble Ticket (TT) systems.

The Aion BRE component must then:

- Recognise that an incident has occurred which requires processing
- Access the necessary data from the JNDMS database to process the incident, compute the damage and update the defensive posture
- Store the results in the JNDMS database.

The primary method of access to the DSS is through the JNDMS Services. The JNDMS Services exports web service interfaces that define the interaction with JNDMS. The JNDMS Services interacts with the DSS through a Web Services interface (SOAP).

The description of the rules themselves is AION specific.

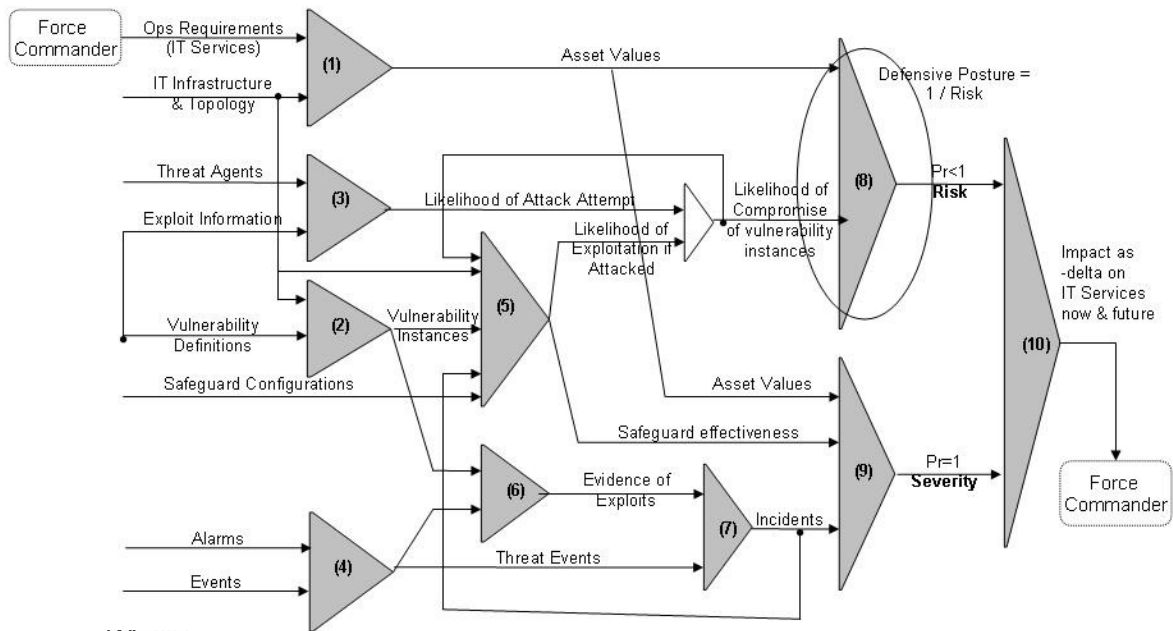
3.7.4 Data Storage

The correlated information, the severity assessments and the defensive posture elements, along with the original rules are stored in the Data Warehouse subsystem. The Data Warehouse contains DSS knowledge (see section section 3.5) continually updated with timestamped severity assessments and incident data generated by the DSS. Severity assessments are stored as attributes of the incident and the vulnerability instance tables.

Aion BRE also has its own repository where all rules are saved and maintained.

3.7.5 Situational Awareness model

The DSS main task is to assess the impact of incidents and vulnerability instances with the objective of providing Situational Awareness (SA). The incidents or vulnerability instances that surpass a predefined threshold of importance are brought to the immediate attention of the user. Figure 3-6 summarizes the model used by the DSS to provide SA in JNDMS.



Where:

- | | |
|--|---|
| 1. Asset Value Assessment | 6. Correlating Events |
| 2. Identifying Vulnerability Instances | 7. Incident Recognition |
| 3. Threat Assessment | 8. Risk Assessment / Defensive Posture Assessment |
| 4. Identifying Events | 9. Severity Assessment |
| 5. Zone & Safeguard Assessments | 10. Providing SA |

Figure 3-6: Situational Awareness Model

3.7.5.1 Asset Value Assessment

Asset Valuation is handled using a combination of two mechanisms; the first is a “static” evaluation of the relative value of assets in terms of their “human attributed” value to military operations. The second is a value dynamically calculated by the DSS at the time of an event that causes damage (impact) to assets which considers the current state of assets, redundancies, and so on.

Static asset valuation is a simple combination of the relative importance of the asset to an operation (OPASSET.IMPORTANCE [essential, veryimportant, important, useful, not applicable] as defined in the JDW) and the relative value of the operation itself (OPERATION.OPERATIONPRIORITY [critical, high, medium, routine, exercise])⁵. These factors are combined to produce a numeric asset value score, stored as ASSET.IMPORTANCE in the database, and used as a weighting factor in the dynamic asset valuation performed during “run time”. Dynamic assessment involves the processing of events, and is described in subsequent sections (for example, see sections 3.7.5.3 and 3.7.5.5).

Asset Valuation

The value of an asset to an operation is determined by how important a service which requires that asset is to an operation. An operation is defined by the services it requires. A service is defined as being essential, very important, important, or useful to an operation, using the same scale as opasset importance. The terms on this scale are then assigned numeric values as shown⁶:

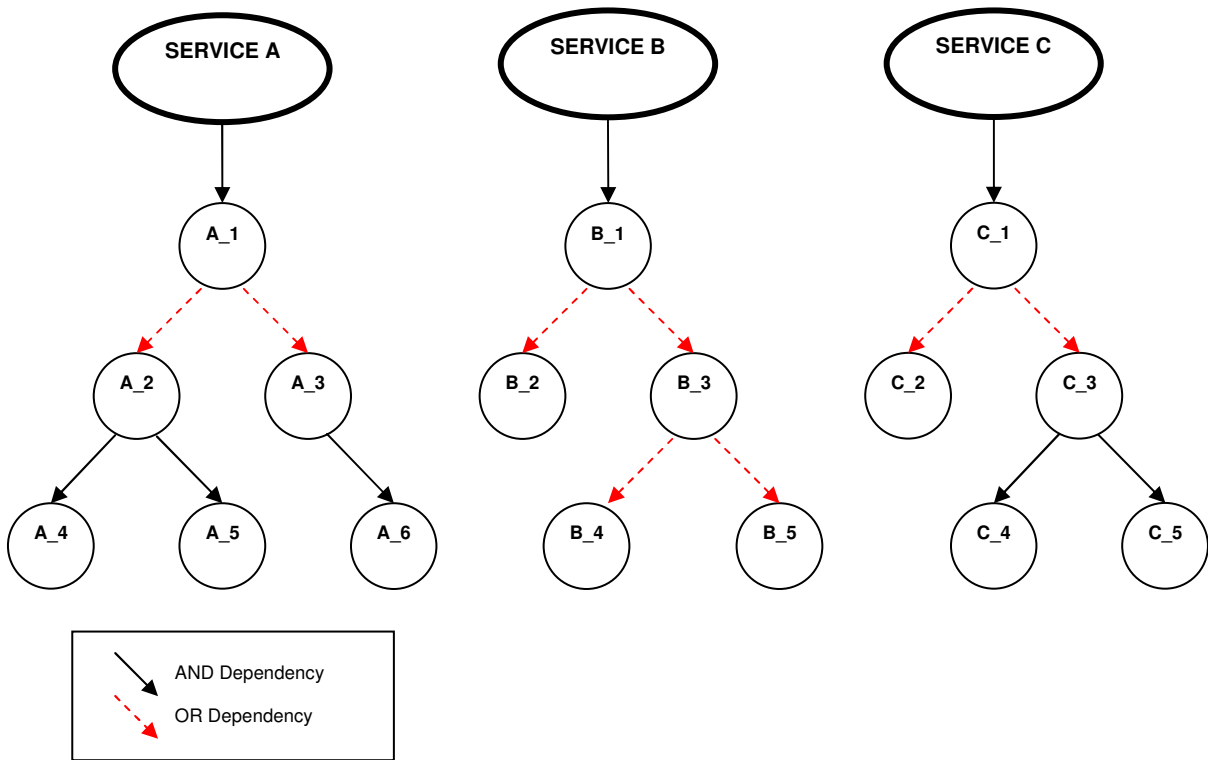
Service Importance	Asset Value
Essential	10
Very Important	8
Important	5
Useful	2
Not Applicable	1

⁵ JSS\DSS\DSSOperation.java. [Note that revision 6117 of JNDMS does not use OPERATION.OPERATIONPRIORITY when performing static asset valuation]

⁶ JSS\DSS\ServiceDependencyGraph.java

A service relies on a combination of hardware and software assets within a dependency chain in order to function. The asset value assigned to these hardware and software assets is determined by the importance of the service to the operation and the dependent and redundant relationships within the dependency chain.

The diagram below shows how asset values are assigned to assets under three different services, each defined as either: essential, important or useful:



Displaying Operation Risk/Impact GYR

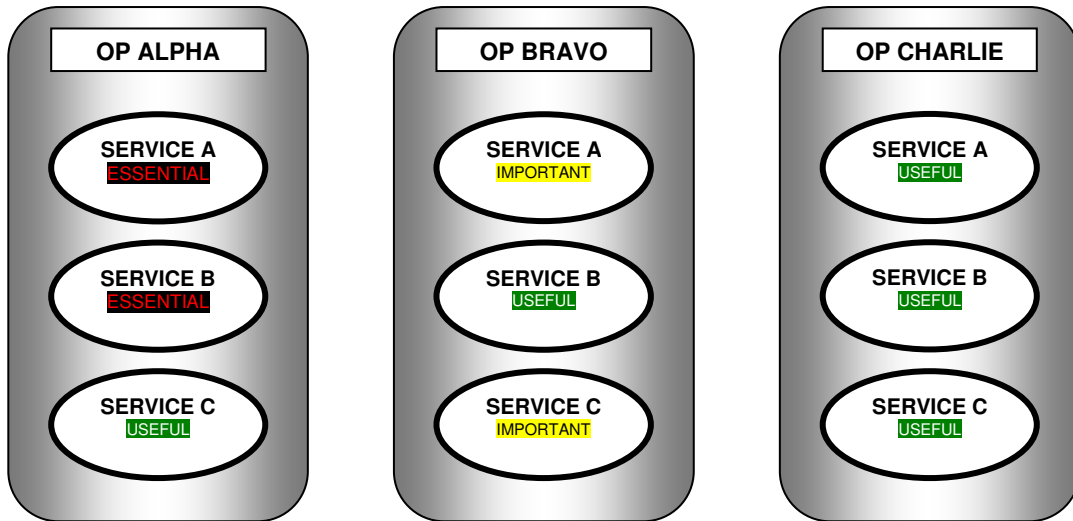
An operation will display a risk value of green, yellow or red, depending on which services are at risk of being impacted, and to what degree the assets within these services are at risk. A predefined threshold for green, yellow and red values will be established.

Colour Code	Value Range ⁷
Green	0 – 2
Yellow	2 – 7

⁷ {Under Review}

Red	7 – 10
-----	--------

Consider the following operations. Each operation requires the three previously defined services, however, each operation places a different importance value on each of the services.



Scenario 1:

Asset A_6 is DOWN → SERVICE A is now 50% impacted

OP ALPHA	OP BRAVO	OP CHARLIE
<ul style="list-style-type: none"> • A_6 asset value = 5 • SERVICE A total impact value = 5 • OP ALPHA total impact value = 5 	<ul style="list-style-type: none"> • A_6 asset value = 2.5 • SERVICE A total impact value = 2.5 • OP BRAVO total impact value = 2.5 	<ul style="list-style-type: none"> • A_6 asset value = 1 • SERVICE A total impact value = 1 • OP CHARLIE total impact value = 1
YELLOW	YELLOW	GREEN

AND Asset A_4 is DOWN → SERVICE A is now 100% impacted

OP ALPHA	OP BRAVO	OP CHARLIE
<ul style="list-style-type: none"> • A_4 asset value = 5 • SERVICE A total impact value = 10 • OP ALPHA total impact value = 10 	<ul style="list-style-type: none"> • A_4 asset value = 2.5 • SERVICE A total impact value = 5 • OP BRAVO total impact value = 5 	<ul style="list-style-type: none"> • A_4 asset value = 1 • SERVICE A total impact value = 2 • OP CHARLIE total impact value = 2

RED	YELLOW	GREEN

Scenario 2:

Asset B_1 has multiply critical vulnerabilities and is accessible from the Internet → SERVICE B is 99% at risk of being compromised.

OP ALPHA	OP BRAVO	OP CHARLIE
<ul style="list-style-type: none"> • B_1 asset value = 10 • SERVICE B total risk value = 9.9 • OP ALPHA total risk value = 9.9 	<ul style="list-style-type: none"> • B_1 asset value = 2 • SERVICE B total risk value = 1.98 • OP BRAVO total risk value = 1.98 	<ul style="list-style-type: none"> • B_1 asset value = 2 • SERVICE B total risk value = 1.98 • OP CHARLIE total risk value = 1.98
RED	GREEN	GREEN

Summary of Results

This method of asset valuation and displaying operation risk will easily allow an operator of JNDMS to quickly assess the risk/impact to an operation. It should be noted that no matter how many “useful” services are impacted or at risk, there will be no increased impact or risk to the operation status. The only services that have the ability to change an operation status to red is a critical service. The highest value that an important service can set the operation status to is yellow. No number of important service outages will set the operation status to red.

3.7.5.1.1 Asset Risk Outside of an Operation

Although JNDMS calculates risk for operations and opassets, a risk value for an asset is calculated, independent of operational need, as an intermediate step in order to facilitate subsequent risk calculation for operations and opassets. This asset risk corresponds to the probability of compromise for that asset. Probability of compromise is calculated as the product of probability of attack and probability of exploit success. These probabilities are stored in the **ASSETRISK** database table. See the Java **RiskRYG** class method named **calcAllAssetRisk()**, which does the following:

- Refreshes zone information.
- Iterates over all hardware assets in the topology graph, doing the following for each one:
 - Calls the doAssetRisk() method, which does the following:
 - Calls the calcRisk() method, which does the following:
 - Calculates $p_{\text{Compromise}} = p_{\text{Attack}} * p_{\text{ExploitSuccess}}$:
 - Calculates pExploitSuccess by calling the RiskUtil.calcPExploitSuccess() method, which does the following:
 - Retrieves a list of INCIDENT.DSSRISK for all active INCIDENT database records with this affected asset (pExploitValues) and calculates a base pExploitSuccess
 - Calculates final pExploitSuccess based on vulnerabilities, safeguards, and some weighting factors, as described in section 3.7.5.5
 - Updates the PEXPLOITABILITY columns of the ASSETRISK database record for the asset
 - Calls the RiskUtil.calcPAttack() method, which does the following:
 - Calculates pAttack as a weighted sum of pAttackLocal, pAttackLocalZone, pAttackRemote and pAttackRemoteZone
 - Updates the PATTACK columns of the ASSETRISK database record for the asset
 - Returns the probability of compromise, therefore asset risk equals probability of compromise
 - Updates the DSSEXPLANATION database record for the asset
 - Updates the RISK columns of the ASSETRISK database record for the asset.

3.7.5.1.2 Example of Asset Value Calculation from Development Dataset

The development dataset contains a network of linked services that provides an environment for examining the calculation of asset values. This section steps through the calculation of asset values for some key service and hardware assets involved in the MARPAC Ops operation. The format of this section is intended to facilitate tracing through this example in an active JNDMS development environment.

Note that the term "service" is used differently in the JSS and the JDW. In the JDW, a service is an asset with a type of either 'network service' or 'capability'. The development dataset holds several network services, but no capabilities. In the JSS, a service is effectively an opasset, as stored in the DSSService class. Note that there does exist a DSSOpAsset class, but it is not widely used in the JSS, other than for some initial data loading.

Figure 3-7 shows the network of linked services upon which the MARPAC Ops operation depends. The figure includes service ID numbers corresponding to the ASSET.SERVICEID values for the corresponding assets in the development dataset. The figure also includes the ASSET.IMPORTANCE asset values as calculated in this example. Note that ASSET.IMPORTANCE values are only written to the database for hardware assets. Asset values for service assets are held in memory for internal processing by JSS.

DN0678
Issue 4/2: 06 February 2012

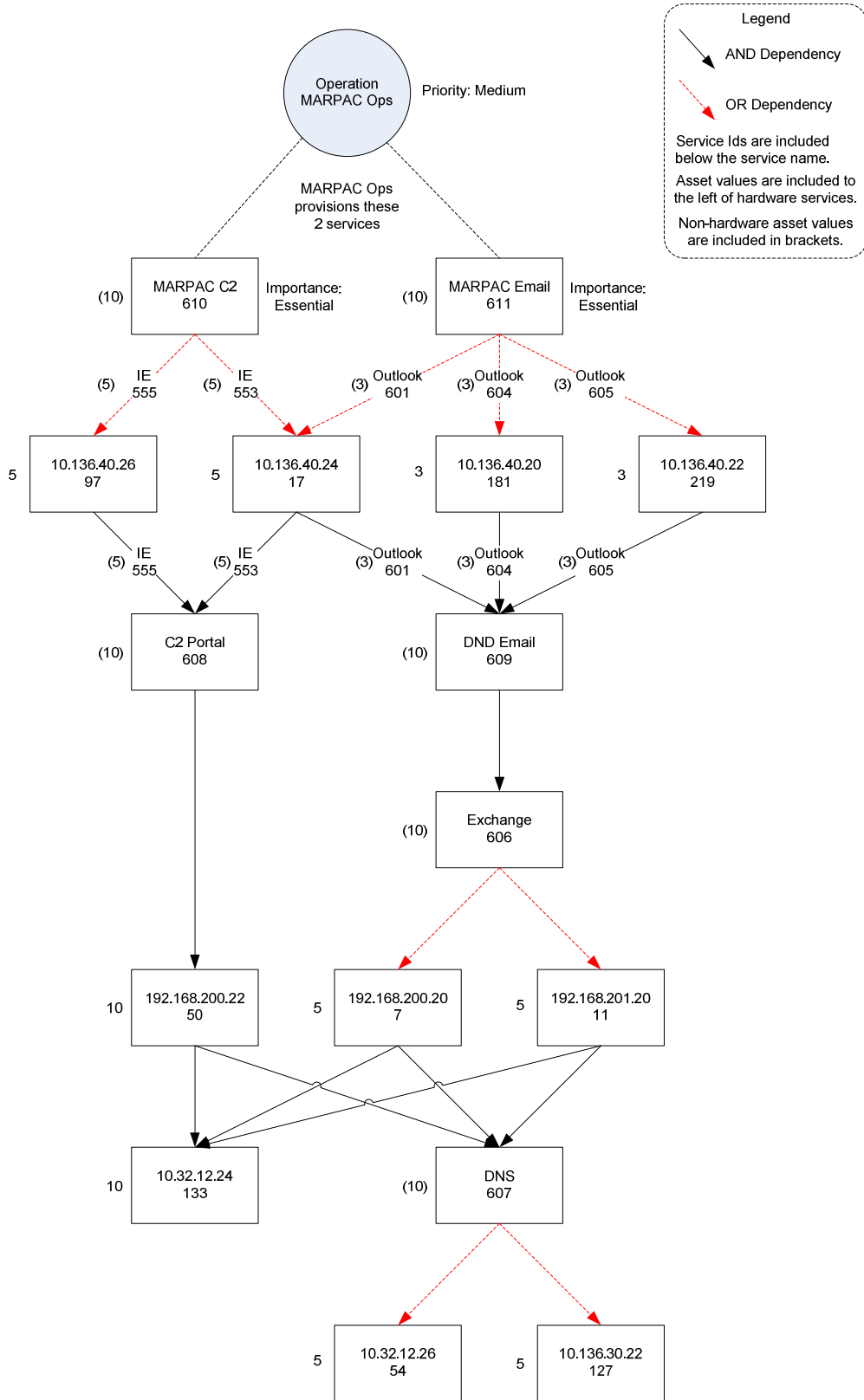


Figure 3-7: MARPAC Ops Service Dependencies

MARPAC Ops provisions only 2 services: MARPAC C2 (id=610) and MARPAC Email (id=611). The presence of the other assets in this example is derived by JSS based on dependency and redundancy information. The process of calculating asset values begins in the `init()` method of the `ServiceDependencyGraph` class, which does the following:

- Retrieves provisioned services for the operation: MARPAC C2 (id=610) and MARPAC Email (id=611).
- Records both services in various internal members, including adding vertices in dependency graph
- Iterates over the 2 services, and for each one, does the following:
 - Builds a "service tree" in order to determine all dependency and redundancy relationships, as follows:
 - Retrieves dependent services. For MARPAC C2 there are none.
 - Retrieves redundant services. For MARPAC C2, these are the instances of Internet Explorer installed on hosts 10.136.40.26 (id=555) and 10.136.40.24 (id=553).
 - For each dependent and redundant service, add vertices and edges accordingly
 - Builds the service tree recursively starting at the new service, thereby eventually building up the entire service dependency tree as shown in the figure
 - Store the service's `OPASSET.IMPORTANCE` value (essential=10) in the `DSSService.assetServiceValue` member
 - Call the `doAssetValuation()` method on the service (see below for a walkthrough of this method)

The `ServiceDependencyGraph.doAssetValuation()` method for MARPAC C2

- Starting at the MARPAC C2 service, traverses the service dependency graph using a `BreadthFirst` iterator named `servDeplerator`.
- The iterator starts at the MARPAC C2 service, and adds its `serviceAssetValue` (10) to its own `DSSService.assetValueToService` hash.
- The iterator continues with the "Internal Explorer on 10.136.40.26" service (id=555).
- Retrieves links to other services that depend on this service: only MARPAC C2 in this case.
- For each linked service, retrieves and sums that service's `assetValueToService`, which is 10 for MARPAC C2, and divides the value by the number of redundant services (2 in this case), for a total of 5.
- In the `assetValueToService` hash of "Internal Explorer on 10.136.40.26" (id=555), add an entry for the MARPAC C2 service, effectively stating that "Internal Explorer on 10.136.40.26" has a value of 5 to MARPAC C2.
- The iterator continues with the "Internal Explorer on 10.136.40.26" service (id=553), performing similar steps.
- The iterator continues with the 10.136.40.26 hardware service (id=97), performing similar steps, as well as the following steps:
 - For hardware services only (as in the case of 10.136.40.26), write the `assetValueToService` to the `DSSService.simAssetValue` member, making sure it does not exceed the asset value of the top level service (MARPAC C2 in this case).
 - For hardware services only, set `DSSService.serviceAssetValue` to the sum of all values in the `assetValueToService` hash (only one value in this case: 5).
- The iterator continues through the entire tree until it finishes with the DNS redundant services 10.32.12.26 (id=54) and 10.136.30.22 (id=127).

Consider the following rules when examining the remainder of the dependency tree in the figure:

- In cases where a service supports more than one service, the asset value of the service equals the sum of the values of the supported services, but it is capped at the value of the top-level service. For example, the asset value of C2 Portal (id=608) is calculated to be 10, the sum of the

DN0678
Issue 4/2: 06 February 2012

2 Internet Explorer software instances (asset value=5 each) that depend on C2 Portal. As an additional example, consider the 10.32.12.24 service (id=133):

- When calculating the asset value for MARPAC C2, 10.32.12.24 is assigned an asset value of 10 because the value of 192.168.200.22 is 10, and the 192.168.200.20 and 192.168.201.20 services have not been processed yet.
- When calculating the asset value for MARPAC Email, a value of 20 (10 for 192.168.200.22 and 5 for each of 192.168.200.20 and 192.168.201.20) is calculated for 10.32.12.24, but the value is capped at 10 because 10 is the value of the top-level MARPAC Email service.
- An asset value will only be assigned to a service if the value is greater than the service's current asset value. For example, consider the 10.136.40.24 service (id=17):
 - When calculating the asset value for MARPAC C2, 10.136.40.24 is assigned an asset value of 5 because the value of "Internet Explorer on 10.136.40.24" is 5
 - When calculating the asset value for MARPAC Email, the asset value for 10.136.40.24 is calculated to be 3.3 because the value of "Outlook on 10.136.40.24" is 3.3, but this new value is lower than the current value, so the current value is kept.

3.7.5.2 Identifying Vulnerability Instances

Vulnerability assessment is handled as follows; Vulnerability Definition events are used to compute Vulnerability Instances on given assets based on matching installed software to affected software versions. Vulnerability Scans generate Vulnerability Instances on affected assets as well. Safeguards (in this case “patches” and the like as stored in the SAFEGUARDPROTECTION table) are used to “mitigate” these. Each “non-safeguarded” Instance will contribute to a P(exploit success) as per the DSS Risk methodology.

There are two distinct sources of vulnerability instances:

1. COTS vulnerability scanners, such as Nessus or eTrust
2. DSS also provides the ability to correlate vulnerability definitions and IT asset configurations to discover vulnerable assets. For example, it may not always be possible to run a new scan when a vulnerability definition is discovered. The analyst will want to query the database for vulnerable assets.
 - DSS will correlate the IT assets with the versions of vulnerable software. The vulnerable software and versions are stored with each vulnerability definition.

Some vulnerabilities will, of course, be discovered by both. They will be tagged with a different timestamp and different sensor source. Updates to the vulnerability instances are generally triggered when a new vulnerability definition is created or modified, or a new asset is created or modified.

3.7.5.3 Threat Assessment

The threat assessment currently looks at threats in four areas and calculates a probability of attack value (pAttack). This probability of attack is used by other processes such as risk assessment.

The four areas examined are:

1. Local threats. This represents threats that target the asset from within the local zone. It is assumed that threats within the local zone have direct access (no perimeter safeguards) to the asset being processed.
2. Local zone threats. This represents threats that exist in the local zone, but don't target the asset in question directly. This is done so that threats that may not be targeting the asset, but nevertheless represents an increase in risk are included in the threat assessment. Threats of this type may include compromises that can be used to spread the threat or to target new hosts.
3. Remote threats. This represents threats that target the asset directly from a remote zone. This will include an assessment of paths between zones and will also include possible filters in zone borders that would prevent access.

4. Remove zone threats. This represents more generalized threats in remove zones. This is similar to the local zone threats; however paths between zones and zone borders are taken into account.

Probability of Attack⁸

The probability of attack is calculated for each asset to include the threats criteria identified above. For each of the categories above the number of attacks is calculated based on the current events or incidents. For example the 'local threats' number of attacks is all attacks in the same zone as the asset where the target is the asset. For remote attacks it would be the number of attacks that have a valid path through zone borders.

In each case the probability of attack is calculated as:

$$pAttack = 1 - 2^{(-numAttacks)}$$

The probability of attack for each of the above domains is combined using user configurable weights. These weights allow the overall threat assessment to be adjusted. The values are combined as follows:

```
pAttack = ZONE_THREAT_WEIGHT * pAttackLocal;  
pAttack = pAttack + (1-pAttack) * ZONE_LOCAL_WEIGHT * pAttackZone;  
pAttack = pAttack + (1-pAttack) * ZONE_REMOTE_THREAT_WEIGHT * pAttackRemote;  
pAttack = pAttack + (1-pAttack) * ZONE_REMOTE_WEIGHT * pAttackRemoteZone;
```

⁸ Applicable routines are found in RiskUtil.java class called from the RiskRYG.java which is initialized soon after the DSS services start.

3.7.5.4 Identifying Events

The alarms and events are identified outside of the DSS and addressed by the JSS, SIM or EIM subsystems.

3.7.5.5 Zone and Safeguard Assessments

CVSS Influence

JNDMS' approach to assessing the vulnerability risk is derived from the National Infrastructure Advisory Council's (NIAC) CVSS. CVSS was created in 2003 to provide a composite score representing the overall severity and risk of a vulnerability. CVSS is a modular approach with three distinct groups that combine the characteristics of vulnerabilities: the base metric, the temporal metric and the environmental metric. In addition to CVSS metrics, JNDMS calculates a base probability of exploit success by examining the risk introduced by any relevant events active on the system.

The "Impact of the Exploit" and the "Likelihood of an Attack" are derived from the CVSS base and temporal metrics. The CVSS environmental metric is not currently used; although it is represented by the "Value of the Asset" for the organisation.

The base metric captures the characteristics that are intrinsic to a given vulnerability. The value is the same in any environment and does not vary with time. The base metric also captures the "Impact of the Exploit" by measuring typical impact on Confidentiality, Integrity and Availability (is it complete, partial or none). The CVSS temporal metric captures characteristics of a vulnerability that change with time.

This stage of assessment uses a factor based on the vulnerabilities, and then looks at active safeguards for each vulnerability.

The first factor identifies⁹ a user configurable curve that indicates that vulnerabilities are more likely to have an exploit available as time goes by:

$$p\text{ExploitAvail} = 1 - 2^{(\text{Exploit_Aging} \times \text{vulnAge})}$$

We then see if we have a CVSS score available (where EXPLOIT_CVSS_x_WEIGHT are configurable parameters):

```
If CVSSscore >= 0 and CVSSscore <= 3.5
    vWeight = EXPLOIT_CVSS_LOW_WEIGHT * cvssScore
Else if CVSSscore > 3.5 and CVSSscore <= 7
    vWeight = EXPLOIT_CVSS_MED_WEIGHT * cvssScore
Else if CVSSscore > 7 and CVSSscore <= 10
    vWeight = EXPLOIT_CVSS_HIGH_WEIGHT * cvssScore
Else
    vWeight = EXPLOIT_CVSS_NONE_WEIGHT
Endif
```

⁹ This is implemented in the *calcVScore* method of the *RiskUtil.java* class.

Our final vulnerability weighting factor is based on the above factors:

$$vScore = pExploitAvail * vWeight$$

We then must identify which of the vulnerabilities have been safeguarded and only include the vulnerability factors (vScore) for exposed vulnerabilities. Our final assessment is based on our current knowledge of the vulnerabilities and available safeguards. This will give us a probability of exploit success, calculated with a user configurable value as¹⁰¹¹:

$$pExploitSuccess = 1 - 2^{-EXPLOIT_SUCCESS_WEIGHT \times totalVScore}$$

3.7.5.6 Correlating Events

Event correlation is broken into two classes “causal correlation” and “coincident correlation”. Causal correlation refers to the case where one event is determined by the DSS as the cause of another event, and this is tracked through a “parent ID” attribute on the “child” event(s).

The root cause of events can be determined by two methods. In the first case we will examine existing events and determine that one is likely the cause of another. In current process the event pairs of a denial of service and an outage are looked at to see if the denial of service has caused the outage.

In the other case the DSS itself will create the child events. This is the case when an outage causes communication issues or breaks service paths. The DSS will make use of the network topology, as well as the dependent and redundant tables to establish the full impact of the given outage (see section 3.7.6 for details).

The DSS must also maintain its expected parent ID in the case where outages are resolved. When an outage is resolved the service dependency model is examined to determine if this will result in the restoration of any other assets. If any of the resolved outages had been identified as a parent (root cause) of another incident, then we must determine if a new root cause would be appropriate. The service dependency model is examined again to find incidents that would prevent the restoration of the given asset and the parent id of the service path incident will be set to the identified outage.

The next class of correlation is coincident correlation where two events are marked by the DSS as coinciding for a given attribute – i.e. event A was “close” to event B (i.e. geographic correlation) or a specific attribute of event A “matched” the corresponding attribute of event B (i.e. IP_down event affecting an asset of type “satellite link” during an active “sunspots” event).

Attributes currently examined as part of coincident correlation include location, target IP, source IP, vulnerability, sensor, and zone.

¹⁰ This is implemented in method *calcPExploitSuccess* of the *RiskUtil.java* class.

¹¹ EXPLOIT_SUCCESS_WEIGHT is a configurable parameter and totalVScore is the sum of the vScores of the current service's vulnerabilities.

3.7.5.7 Incident Recognition

While there is generally some debate as to the difference between what constitutes an “event” and what constitutes an “incident” the DSS adopts a pragmatic default position; if an event sent to the DSS is determined to cause impact, it is marked as an incident, otherwise it left as an event. This means that events that cause “risk” (potential future impact) are still just events (but since they can be ranked by a risk score, they are easily prioritized against other events).

When determining if impact has occurred, the DSS handles grey areas by means of a (user configurable) threshold of certainty – the event probability of success. One example of this is an “exploit event” which targets a vulnerable asset, and there is a match on the exploit type and vulnerability type – the DSS does not have “direct evidence” of impact, but since the probability of success is about the default threshold of certainty it is marked as an incident.

An incident may have an impact on Availability (denial of service), Integrity (unauthorized modifications), and/or Confidentiality (unauthorized disclosure). Incidents that do not cause damage may be signs of threat (unsuccessful attacks) and must be monitored and considered when assessing the current and potential threat level. These events, such as reconnaissance events, are often part of a threat vector.

Table 17: List of JNDMS Incident Types

ID	Incident Types	Source	Impact			Attack Attempts
			C	I	A	
1	Outage (including Service Releases)	TT EIM	None	None	Total	
2	Degradation	EIM	None	None	Partial	
3	Network Traffic Anomalies	EIM SIM (ID 1)	None	None	None	x
4	Hosts Discrepancies (Discovery vs. Baseline)	EIM	None	None	None	x
5	Unauthorized Access - Root Access	DSS	Total	Total	Total	
6	Unauthorized Access - User Access	DSS	Partial	Partial	Partial	
7	Unauthorized Access – Access Unknown	SIM (ID 3)	Partial (or Total)	Partial (or Total)	Partial (or Total)	
8	Access Attempt	SIM (ID 2A)	None	None	None	x
9	DoS	SIM (ID 6)	None	None	Total (or Partial)	
10	Policy – TRANSEC	TT	Total	None	None	

DN0678
Issue 4/2: 06 February 2012

ID	Incident Types	Source	Impact			Attack Attempts
			C	I	A	
11	Policy – Inappropriate Storage	TT	Total	None	None	
12	Policy – Misconfiguration	TT	None	None	None	x
13	Policy – Unauthorized Use	TT SIM (ID 12)	Total	Total	None	x
14	Policy – Unauthorized IT Asset	TT DSS	Partial	None	None	x
15	Policy – Other	TT	?	?	?	x
16	Malicious Logic – Virus	SIM (ID 16)	Total	Total	Total	
17	Malicious Logic – Trojan	SIM (ID 16)	Total	Total	Total	
18	Malicious Logic – Worm	SIM (ID 16)	Total	Total	Total	
19	Malicious Logic – Other	SIM (ID 16)	Total	Total	Total	
20	Exploit Attempt – Likely Successful	SIM (ID 9)	Total	Total	Total	
21	Exploit Attempt – Unlikely Successful	SIM (ID 9)	None	None	None	x
22	Reconnaissance – Probe	SIM (ID 13)	None	None	None	x
23	Reconnaissance – Port Scan	SIM (ID 14)	None	None	None	x
24	Reconnaissance – Network/Service Scan	SIM (ID 17)	None	None	None	x
25	Reconnaissance and Disclosure	SIM (ID 5)	None	None	None	x
26	Evasion Attempt	SIM (ID 8)	None	None	None	x
27	Insecure Behaviour	SIM (ID 11)	None	None	None	x
28	Physical (Nature, Environment, Man-made)	TT, JNDMS	None	Partial, Total	Partial, Total	

3.7.5.8 Risk Assessment / Defensive Posture Assessment

The defensive posture assessment is a key part of the research within JNDMS. As such this is targeted during each development cycle to develop and formalize how this assessment will be carried out and presented. The initial design addressed key inputs and a first take at some of the fundamentals, i.e., capturing the vulnerabilities, the safeguards and their impacts on Confidentiality, Integrity, Availability and Authentication.

The draft technical memorandum “CND SA Information Requirements” [R-5] describes the defensive posture as the search for “the set of threat vectors, which are capable of exploiting vulnerabilities in the Information Technology Infrastructure (ITI) resources supporting mission critical IT Services for which there are insufficient safeguards.”

The final risk value combines the results of the safeguard assessment, the asset valuation and the threat assessment. The values are combined as follows for each asset¹²:

```
pCompromise = pAttack * pExploitSuccess  
risk = pCompromise * Asset Importance
```

3.7.5.9 Severity Assessment

The severity assessment is based on the asset value assessment. The value of the assets impacted by a given event, modified by the impact of the event, sets the severity of that event.

3.7.6 Overview of DSS processing stages

The Decision Support System (DSS) has two principle stages of operation; initialization and event processing.

¹² This is implemented in the *calcRisk* method of the *RiskRYG.java* class.

3.7.7 Initialization

During Initialization the DSS reads in data from the JNDMS Data Warehouse (JDW) about the Computer Network Defence (CND) environment to build a “knowledgebase” of in-memory state models to be used both as a reference and to cache intermediate results for event processing by the rules.

Initialization is called when the DSS is first instantiated by JNDMS during system start up, and during various cases when “re-initialization” is required, such as when data has been bulk loaded into the database, or when user entered changes to the database have occurred without the DSS having received the events that would normally be associated with those changes.

Initializing the DSS involves a number of steps:

- Building the DSS Knowledgebase
 - Processing the Topology data into the IT Topology model
 - Processing Service relationship data into the Service Dependency model
 - Processing the IT Topology and Service Dependency models to produce a Service Path model
- Processing Military Operations data, Vulnerability data, Software inventory data, Safeguard data, and Incident data (if present) to perform
 - Asset Valuation
 - Initial Risk Assessment
- Processing Zone data (if present) to initialize Aion DSS and perform additional Risk Assessment
- Starting internal “Risk timer” to generate periodic risk assessment events

3.7.7.1 DSS Knowledgebase

The in memory data models representing state of CND environment are used for a number of reasons including:

- Specialized data structures/ state models allow the use of associated sets of efficient “tools” for searching/ processing state data
- Efficient (rapid) access of data inside the working memory of the rules (rather than needing to access the same data in a relational structure on a system over the network)
- Express the collective state of various discrete aspects of the CND environment in a way which eases human understanding (compared to the equivalent structure stored in a relational schema)

Key attributes of the in-memory data models are cached to JDW after initialization to allow accurate rebuilding of state in the case of a system restart of a “live” CND environment (which has active Incidents, current Risk or Impact, and so on).

The key models in the DSS working memory are:

- IT Topology model
- Service Dependency model
- Service Paths model

3.7.7.1.1 IT Topology Model

The DSS IT Topology model is built in memory as an undirected graph based on the asset and link tables in the JDW. When a node on the graph (an asset such as a host, switch, or router) becomes unavailable, the DSS removes the edges to its neighbours and connectivity is “broken”. When the asset is once again available, the edges (and connectivity) are “restored”. When represented visually, this graph is equivalent to a “logical” network diagram (independent of layout). Additionally nodes in the graph track various attributes of the ASSETs for use by the DSS, including availability state, associated IP addresses, and so on.

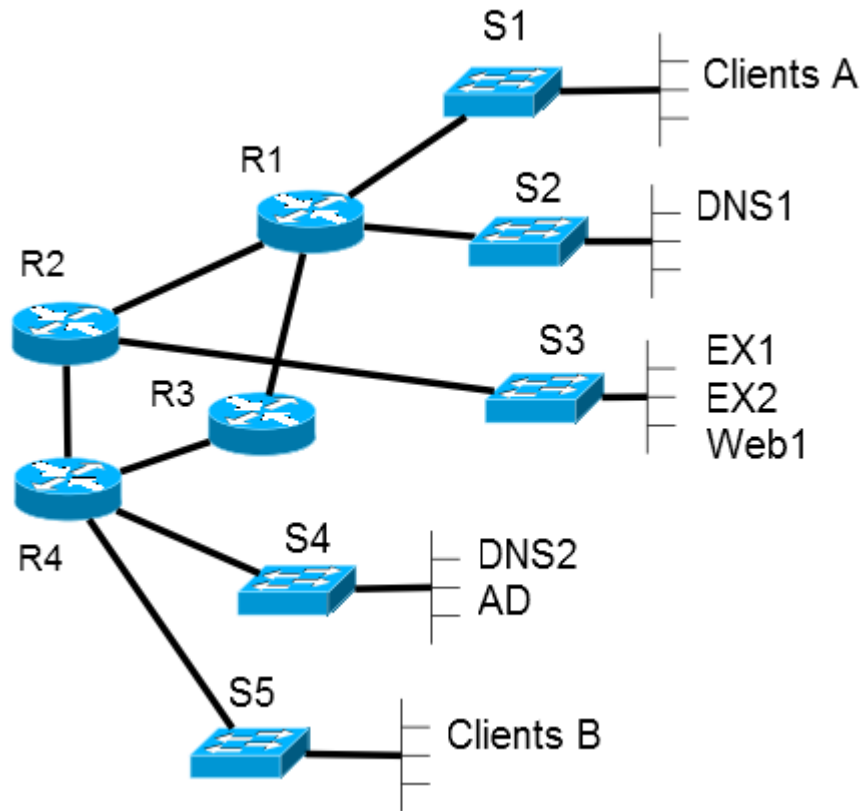


Figure 3-8: IT Topology Model

3.7.7.1.2 Service Dependency Model

The DSS Service Dependency model is built in memory as a directed graph based on the ASSET, DEPENDENT, and REDUNDANT data in the data warehouse. When a node on the graph becomes unavailable its “status” is updated, and Boolean logic is applied to determine how the parent(s) are affected, and if they have now become unavailable. Direction of the edges in the graph represents the direction of the “depends on” or “depends redundantly on” relationships. As in the topology graph, the service dependency nodes contain a number of attributes and “meta data” the DSS tracks on assets with client/server relationships such as Service providing state of Assets (such as servers, logical services), Service Path state, Service Dependency state, and so on.

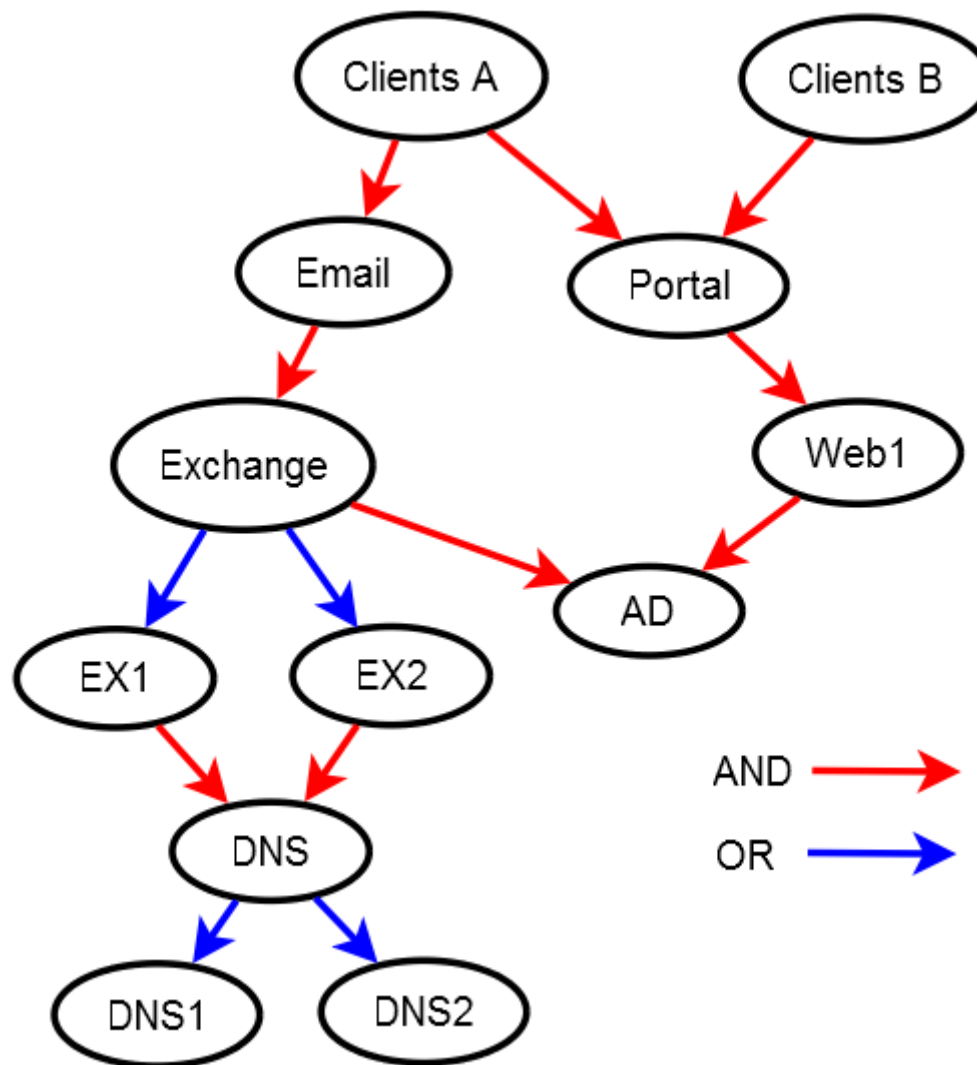


Figure 3-9: Service Dependency Model

Asset Valuation is now being handled through a mix of “pre-computed” and “dynamic” valuation. An “intrinsic” value is computed for each asset at initialization (or data entry time) and factors in OPASSET and OPERATION priorities on that asset. If an INCIDENT then has an impact on that asset, DEPENDENT and REDUNDANT relationships are considered based on the current state of the network to assign a value to that asset which corresponds to the impact that incident has on the “intrinsic” value of all affected assets.

3.7.7.1.3 Service Paths Model

The DSS Service Paths model is built in memory as list of Boolean equations based on a merged understanding of the IT Topology Model and Service Dependency Model. This is represented graphically here for ease of understanding, along with the associated list of Service Paths.

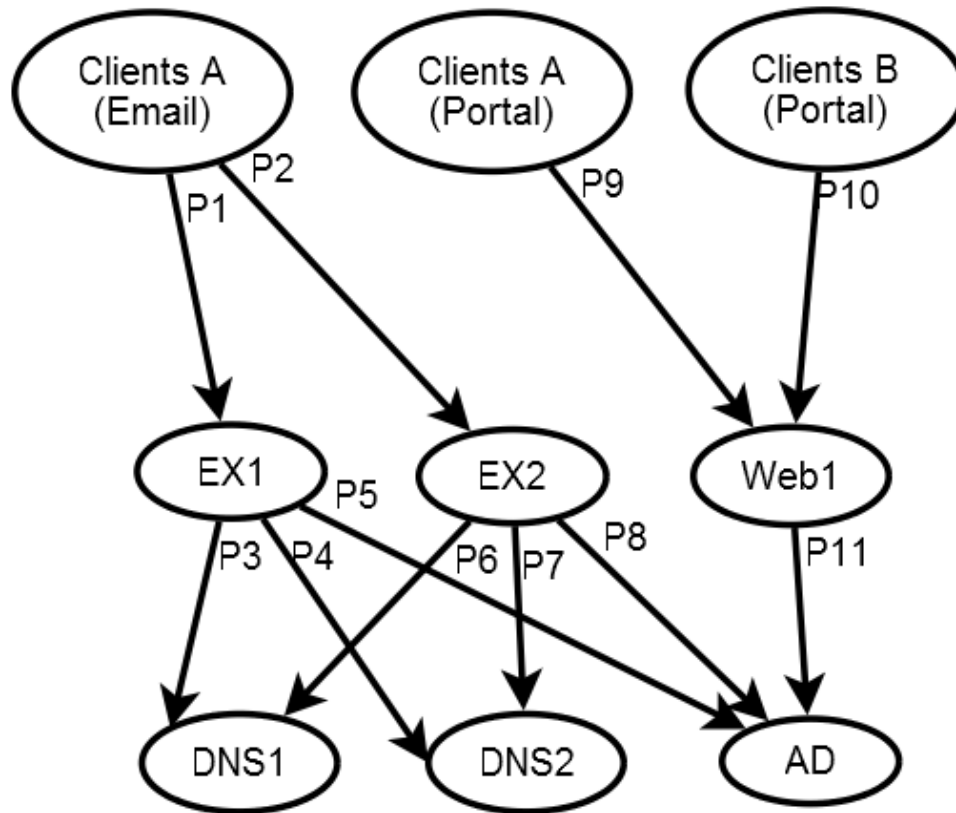


Figure 3-10: Service Paths Graphical Representation

Implied Service Paths

Figure 3-10, above, shows the merging of the physical topology (Figure 3-8) with the service dependency model (Figure 3-9) to show the relationship between physical assets (those that exist in the topology). The relationship is based not only on the specified dependency or redundancy extended to the physical assets, but also the state of the network topology. For example in the above diagram the following would hold:

P1 is available if PathExists(S1, S3) in topology
P2 is available if PathExists(S1, S3) in topology
P3 is available if PathExists(S3,S2) in topology
P10 is available if PathExists(S1, S3) in topology
P11 is available if PathExists(S3, S4) in topology

3.7.7.1.4 Zone Model

A model of the zones is also kept in memory. This model represents a simplified topology model that includes zones and zone borders. This model will identify if there is a path from one zone to another by examining the topology model. Two zones would be adjacent (with a direct path) if they share a common zone border.

This model is used to quickly assess what paths exist from one zone to another and to identify perimeter (border) safeguards that must be addressed.

3.7.7.2 Risk Initialization

At initialization, a Risk score is calculated for each ASSET in the JDW based on factors that affect each asset individually. At a high level this is a combination of the probability of compromise of the asset and the impact the compromise would have (in terms of asset value).

Depending on the CND data available to the DSS, relevant inputs include:

- Probability of attack attempt on the asset
 - History of attack attempts on the asset
 - Attack vectors between the asset and threat agents (safeguards)
 - “Proximity” to assets that are themselves compromised or at risk
 - Asset is dependent on Asset that is at risk
- Severity of Vulnerabilities on the Asset
 - Age of the vulnerabilities (and likelihood of exploits)
 - Potential impact of vulnerabilities

- Probability of Exploitation if Attacked
 - Mitigation of vulnerabilities (safeguards)
 - Severity of Vulnerabilities
- Probability of Compromise
 - Probability of attack attempt
 - Probability of exploitation if attacked

3.7.7.2.1 Risk Timer

The risk assessment can be a time consuming process and has the potential to significantly impact the ability of the DSS to process many events. The risk assessment itself, however, is based solely on the current state. The DSS implements a method (generally called through the `dp_update` interface of the JSS) that will discard the current risk assessment and re-calculate the risk assessment for every asset. The ability of the DSS to separate the risk assessment from the event processing allows the risk assessment to be called asynchronously and can process more than one event at a time.

During initialization the DSS starts an internal “Risk timer” to generate periodic risk assessment “events”. Since Risk is a function of Probability, and Probability is a function of time, the DSS must ensure that various time-dependent probabilities are updated periodically to enable the JNDMS user to view the current risk to assets and operations.

3.7.8 Event Processing

After initialization, the DSS runs in a “steady state” mode of operation where each event delivered to the system is handled as an individual transaction, and processed according to a series of “rules” in the context of both the type of event, and the CND data held in-memory in the DSS Knowledgebase.

3.7.8.1 Event Processing methodology

As each event is received it is processed to update the incident assessment, the event correlation, the threat assessment, vulnerability assessment, the safeguard assessment and the risk assessment.

DSS Explanation

As an aide to the JNMDS user, the DSS generates an “explanation statement” for root cause incidents which provides the supporting evidence for the impact the event has on operations (detailing if other assets were affected due to the loss of communication path or service dependency) and why the severity ranking was assigned (based on the value of affected assets).

3.7.8.1.1 Cascade Impact

JNDMS wishes to provide Situational Awareness of military operational impact at a higher level than what would be provided by a simple “up/down” status on monitored servers. Of particular interest are cases where an asset becomes unavailable which in turn affects the availability of other assets in a “cascade” fashion.

The goal of the cascade impact rules in the DSS is to provide the operational impact of a given outage to JNDMS users to allow both better root cause analysis of service outages and to allow better assessment of the urgency of server/network outages on the services required by Operations.

This approach deals with this problem at the level of the loss of availability of servers (“host” assets which provide IT services) or networks (devices and links which provide connectivity) to clients (users whose operational requirements are specified in the system). The DSS Knowledgebase contains the IT topology model and Service Dependency models of the CND environment and using these references, the DSS can then calculate the “cascade impact” to an Operation of an outage to assets in either the topology or service dependency “chains”.

Given the preceding IT Topology Model, Service Dependency Model, and Service Paths Model the following Cascade Impact Rules are generated.

Clients A (Email) => P1 or P2
EX1 => P5 and (P3 or P4)
EX2 => P8 and (P6 or P7)
Clients B (Portal) => P10
Web1 => P11

When an ASSET (such as a host, switch, or router) becomes unavailable, the DSS queries the IT Topology model to determine if the list of ServicePaths “exist” and assigns a TRUE or FALSE value to each path. The list of Availability Impact Rules is then in turn evaluated using Boolean logic to assess if any of the assets have become FALSE (unavailable).

3.8 Presentation, Visualization and Alerting

3.8.1 Overview

The Presentation, Visualization and Alerting component will present SA data to the users and allow user-control of the JNDMS TD.

The user interaction with JNDMS will be done through a web browser and will be delivered through a portal. The choice of a thin client, using a browser, over possible thick client alternatives was made for several reasons. The use and support of web technologies has grown significantly over the past few years, and the vast majority of COTS products examined support web technologies in one form or another. This common technology allows easier integration of disparate products and allows the portal to collect and display a number of data sources.

The choice of a web browser also eases the deployment of the clients and allows central management of the system. These properties and the wide support for core web standards will ease testing and eventual deployment.

Another factor in this choice is the progress that web interfaces have made in the past few years; especially with the wide spread use of Asynchronous JavaScript and XML (AJAX). The user interfaces are becoming more interactive and responsive which allows for greater flexibility during development.

The views presented in this section will be refined during the development and will represent the views to be developed for the TD.

3.8.2 COTS Selection

This component will be created from a mixture of COTS and custom code, tailored to satisfy the user interface requirements; Oracle for component data management and the Liferay portal. The GIS is based on the ESRI Server GIS technology.

ESRI was chosen because of its broad base of support of GIS technologies, as well as their commitment to server based GIS and web clients. This is central to the JNDMS architecture, and the ESRI Developers Network allows the development team to integrate the components that are required.

3.8.3 Component Design

3.8.3.1 Portal

The portal for JNDMS is built using the Liferay Portal. This is a Java standards based portal (JSR-168) environment that can be used to leverage open standards and wide commercial support.

The portal, as deployed in cycle 3, makes use of the Tomcat servlet container. The Tomcat servlet container is an open source initiative by the Apache Foundation that has gained wide support as a base implementation for many J2EE server environments. The use of the common servlet container for the portal also allows the ESRI map and the JSS web applications to be deployed on the same server where appropriate.

3.8.3.2 GIS

3.8.3.2.1 Overview

This section describes the technologies used to implement the geographic map displays on the web application client. Both server-side and client-side components will be covered.

The GIS functionality is required to display maps and overlay information such as network outages, show vulnerable nodes and possibly show related geographical information, such as current weather information. The following requirements for the GIS system are identified:

1. The baseline maps consist of geographical locations and administrative boundaries. These maps provide geo-references for the display of JNDMS network nodes and links.
2. Normally, the baseline maps are static and there is no need to manipulate baseline map data layers during the JNDMS system operations.
3. The JNDMS data sets, such as network nodes and links locations and the network operation status, are changed dynamically. The map display shall be updated periodically to reflect these changes.
4. All the baseline map data and JNDMS location data shall use a common geodetic coordinate system, e.g., World Geodetic System 1984 (WGS 84).
5. The mapping client application shall have basic viewing capabilities, such as zoom in/out and pan functions.
6. User can query on the network node icons to request further information, such as summary information or links to additional views.

Figure 3-11 illustrates a client server application for the GIS implementation.

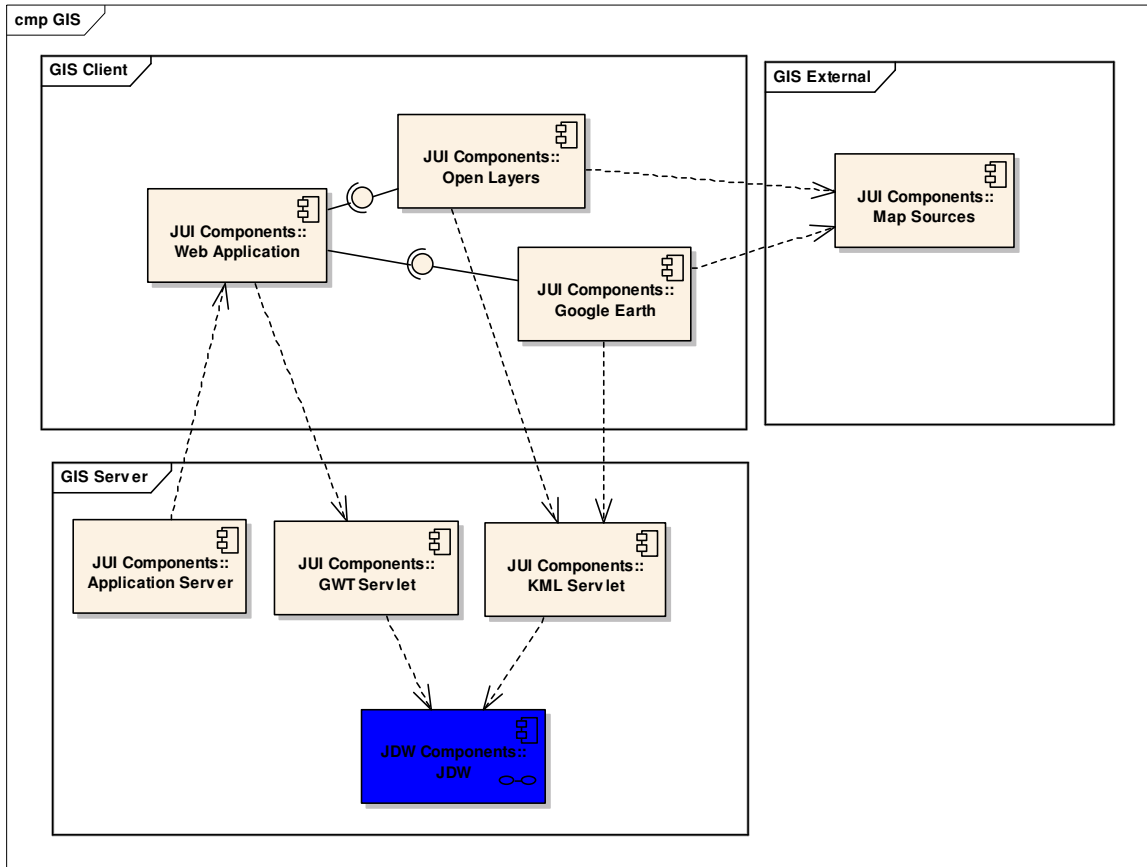


Figure 3-11: GIS Information Flow

3.8.3.2.2 Server-Side Components

3.8.3.2.2.1 Java Web Application Technologies

JNDMS is a Java Web Application. The application servers used are described in this section. While JNDMS is made up of multiple web apps that work together, this document will only discuss the specifics of the JNDMS Portal web application.

3.8.3.2.2.1.1 Tomcat Application Server

The primary application server used for JNDMS is Tomcat. Currently, this is used in the development lab environment, as well as at the client site.

3.8.3.2.2.1.2 Jetty Application Server

Jetty is used by the Google Web Toolkit when you launch the JNDMS Portal in hosted mode during development. This can introduce some inconsistencies between what you encounter during development, and what you encounter when the built Web Application Archive (WAR) is deployed to Tomcat. For example, in GWT Hosted Mode, the web app is deployed by default to the root webapp context. This is usually not the case when deploying to Tomcat. Hosted mode also defaults to non-secure HTTP, rather than HTTPS. These differences can cause some extra work when the code is built and deployed to Tomcat, so make sure to do a sanity test in both environments on a regular basis.

3.8.3.2.2.2 Spring Framework

JNDMS Portal uses a number of features from the Spring Framework, and is therefore dependent upon a number of Spring Framework JAR libraries. The build scripts must be set up to include these in the WAR, otherwise the JAR files must exist on the common classpath for each application server.

This section will discuss the JNDMS Portal configuration of Spring at a high level, so new developers will have a starting point if they need to make changes to Spring. For a detailed Spring Framework user guide, please go to: <http://www.springframework.org/>

Most of the Spring-dependent classes in the JNDMS Portal can be found in the following packages:

- `com.mdacorporation.jndms.jui.server.web`
- `com.mdacorporation.jndms.jui.server.dao`

3.8.3.2.2.1 Spring Configuration

3.8.3.2.2.1.1 XML Configuration Files

Part of the Spring configuration in JNDMS Portal is contained in XML files. Currently, these are packaged in the WAR file structure at:

```
WEB-INF/conf/spring/
```

In the project code, they are located here:

```
resources/web/conf/spring/
```

Refer to Chapter 3 of the Spring Framework Reference for details on the XML configuration file format.

3.8.3.2.2.1.2 Web Application web.xml Configuration

The JNDMS Portal web app has some Spring configuration inside the web.xml file. One mandatory value is the context parameter “contextConfigLocation”, whose value is used to locate the core Spring XML configuration file. The web.xml file should contain this element:

```
<context-param>
  <param-name>contextConfigLocation</param-name>
  <param-value>/WEB-INF/conf/spring/jui-core.xml</param-value>
</context-param>
```

Since we are also using the features of Spring MVC, an instance of DispatcherServlet must be added to our web application. This is done by adding a <servlet> element to web.xml as shown here:

```
<servlet>
  <description>Spring MVC Dispatcher Servlet</description>
  <servlet-name>jui</servlet-name>
  <servlet-class>org.springframework.web.servlet.DispatcherServlet
</servlet-class>
  <init-param>
    <param-name>contextConfigLocation</param-name>
    <param-value>/WEB-INF/conf/spring/jui-servlet.xml</param-value>
  </init-param>
  <load-on-startup>1</load-on-startup>
</servlet>
```

A <servlet-mapping> element must be added to direct any incoming requests to your dispatcher servlet, as shown here:

```
<servlet-mapping>
    <servlet-name>jui</servlet-name>
    <url-pattern>/map</url-pattern>
    <url-pattern>/map/*</url-pattern>
    <url-pattern>/kml</url-pattern>
    <url-pattern>/kml/*</url-pattern>
</servlet-mapping>
```

You must also load an instance of `ContextLoaderListener`, as shown here:

```
<listener>
    <listener-class>org.springframework.web.context.ContextLoaderListener
</listener-class>
</listener>
```

Finally, Spring MVC is capable of using context parameter values as attribute values when creating class instances via the IoC container (mentioned below). JNDMS Portal is using this feature currently, as shown below:

```
<context-param>
    <param-name>databaseConnection</param-name>
    <param-value>@jndms.db.jdbc_url@</param-value>
</context-param>
<context-param>
    <param-name>databaseName</param-name>
    <param-value>@jndms.db.user@</param-value>
</context-param>
<context-param>
    <param-name>databasePassword</param-name>
    <param-value>@jndms.db.password@</param-value>
</context-param>
<context-param>
    <param-name>kmlStyleBaseUrl</param-name>
    <param-value>@jndms.jui.kmlStyleUrlFactory.baseUrl@</param-value>
</context-param>
```

The `@` characters are nothing Spring-specific. These are just placeholder values that are replaced by Ant properties during the build process. All of the values shown above are defined in properties files that are read by the Ant build scripts.

3.8.3.2.2.2 Inversion of Control (IoC)

Inversion of Control (IoC), sometimes referred to as “Dependency Injection”, is a core feature of the Spring Framework. A number of JNDMS Portal classes are managed by Spring MVC and have their dependencies provided by the framework via IoC.

Refer to Chapter 3 of the Spring Framework Reference for details on Inversion of Control.

3.8.3.2.2.3 Spring MVC

The Spring Framework provides a Model-View-Controller infrastructure if you include the Spring MVC supplementary JAR file in your webapp classpath. The core class is the `DispatcherServlet`, which is used to examine incoming HTTP requests and dispatch them to an appropriate controller class. The `KmlController` class, for example, is central to the KML data generation workflow.

Refer to Chapter 13 of the Spring Framework Reference for details on Spring MVC.

3.8.3.2.2.4 Annotated Classes

Many Spring-enabled classes in the JNDMS Portal use annotations to define their properties and dependencies, rather than defining this in the Spring XML configuration files. The advantage is that these properties are all consolidated within the class definition and help the developer understand how the class is being used. The disadvantage is that the configuration is spread out among all the individual classes rather than consolidated in an XML configuration file. There is no right or wrong choice here, but rather a design choice. Most of the JNDMS classes are designed to use the annotation style.

Some of the annotations that we use are:

- `@Component` – Marks class as one that should be managed by the Spring IoC framework.
- `@Repository` – Marks class as a Data Access Object (DAO) class. This is a specialization of `@Component`.
- `@Controller` – Marks class as an MVC controller. This is a specialization of `@Component`.
- `@Resource` – Marks ‘setter’ method as target for dependency injection.
- `@Required` – Used in combination with `@Resource` to make it mandatory.
- `@PostConstruct` – Marks any no-arg method as an initializer method that must be invoked by the IoC framework after all dependency injection is completed. This gives you a chance to add any custom initialization code that you want executed each time an instance of this class is created by Spring.
- `@RequestMapping` – Marks a method as a request handler. See the Spring MVC documentation for details on how this works.

Refer to Chapter 3 of the Spring Framework Reference for details on configuring Spring to recognize annotations.

3.8.3.2.2.5 Properties Files and Context Parameters

Spring has the ability to read in simple attribute values from both properties files and Web Application context parameters, then uses these during the dependency injection process. This is done in a few places in the JNDMS Portal code. Please refer to the Spring Framework Reference to the specifics on how this works.

3.8.3.2.2.3 KML Data Generation

Each time the JNDMS Portal client maps (OpenLayers and Google Earth Plugin) make a request for KML data to the server, a fresh KML document is dynamically created.

3.8.3.2.2.3.1 Retrieving the Requested Data

The JNDMS Portal uses the PageView class as a container for the elements that describe a request. These are primarily the 'focus' table of the query, and zero or more filter conditions. While the non-map portions of the JNDMS Portal use a "generic SQL" scheme to retrieve data, this was insufficient for the KML generation module for performance reasons. The SQL queries for the KML data are generated dynamically, based on the contents of the PageView associated with each request. The SQL is created by this class:

```
com.mdacorporation.jndms.jui.server.dao.GeoMapSQLFactory
```

The execution of the SQL and mapping into result objects is done by this class:

```
com.mdacorporation.jndms.jui.server.dao.GeoMapDaoImpl
```

The classes used to store the results of a SQL query for geo map data are contained in this package:

```
com.mdacorporation.jndms.jui.server.web.model
```

The whole process from receiving a request for KML to sending a response containing a KML document is controlled by this class:

```
com.mdacorporation.jndms.jui.server.web.controller.KmlController
```

The process for converting the SQL query results into a KML document will be described at a high level in the following sections.

3.8.3.2.2.3.2 Creation of the KML Document

3.8.3.2.2.3.2.1 KML 2.2 Schema Classes

Within the overall JNDMS project code base is a sub-project called:

```
mda-ogc-kml-v_2_2_0-schema
```

This was created prior to my joining the JNDMS team, and I had no part in the creation of this sub-project. I believe that it creates a JAR with classes that represent the KML 2.2 schema elements. This JAR is used by the KML generation code to build an object-based representation of a KML document, then marshals the data into a real KML document at the end of the process.

3.8.3.2.2.3.2.2 Creating Placemark Description HTML

The placemarks that are contained in the KML, and subsequently displayed on the maps, show you locations that are related to the PageView in your request. A more detailed description of the placemark will be displayed in a popup bubble if you click on the placemark. The HTML content in the popup is generated by the following class:

```
com.mdacorporation.jndms.jui.server.web.view.DefaultKmlGeometryDescGenerator
```

An instance of DefaultKmlGeometryDescGenerator is used by the KmlController to populate the <description> element of each KML placemark in the KML document.

3.8.3.2.2.3.2.3 KML Styles

KML styles are defined in a separate document whose URL is referenced in each dynamically created KML document. This prevents having to copy the style definitions into each dynamically created KML document. The KML style document for JNDMS Portal is copied to the root of the WAR file by an Ant script during the build process. The master copy is stored here:

```
/resources/kmlStyles.xml
```

3.8.3.2.2.3.2.4 Miscellaneous Issues

There is an issue with the interaction between the Google Earth Plugin and the Jetty Application Server. When the Google Earth Plugin makes a request for a KML document to a webapp hosted by Jetty, the JSESSIONID cookie and/or the URL-embedded jsessionid are lost. The JNDMS webapp will be unable to find the user session and the request is rejected. Jetty is the embedded application server used in the GWT development environment (more on this later).

In order to work around this limitation, I have created a servlet filter, a listener and a request wrapper to fake out the server code into accepting the request. Please take a look at the following classes:

```
com.mdacorporation.jndms.jui.server.web.listener.SessionTrackerListener
```

```
com.mdacorporation.jndms.jui.server.web.listener.GoogleEarthPluginSessionTrackerListener
```

```
com.mdacorporation.jndms.jui.server.web.filter.SessionIdRepairFilter
```

3.8.3.2.3 Client-Side Components

3.8.3.2.3.1 Google Web Toolkit (GWT)

I won't get into a lengthy discussion of all the features provided by the Google Web Toolkit (which I will refer to as GWT from now on), since you can read for yourself on their project website. The one feature of GWT that I have used **extensively** for JNDMS is the ability to "wrapper" JavaScript code within Java native methods. This allows you to execute the application (server and client) in a "hosted mode" development environment where you can debug through much of your Java code – essentially all the code except for the native JavaScript. When it comes time to deploy into a production environment, the GWT framework will translate your GWT-integrated Java code into pure JavaScript. The people at Google claim that the generated JavaScript is optimized and in many cases will run faster than hand-written JavaScript code.

3.8.3.2.3.1.1 Ramping Up on GWT

First of all, you'll need to be able to write JavaScript code if you want to maintain the OpenLayers and Google Earth Plugin wrapper classes used in JNDMS Portal. If you have no experience with JavaScript coding (as I didn't when I joined the project), then you should stop now and go read up on JavaScript. Make sure that you understand how "object-oriented" JavaScript works. Also, make sure that you know about JSON (JavaScript Object Notation), as this is used extensively by the OpenLayers mapping library source code. You don't have to be a JavaScript expert, but you need to understand the core parts.

As far as GWT is concerned, it will save you a lot of headaches if you start by reading (yes, the WHOLE thing!) the following document:

<http://code.google.com/webtoolkit/doc/1.6/DevGuideCodingBasics.html>

Pay close attention to the JSNI (JavaScript Native Interface) section. JSNI is what I used to wrapper the OpenLayers JavaScript libraries for use in JNDMS Portal. You won't need to immediately use some of the other features that you read about in the dev guide, but it is important to know that they exist in case you need them in the future.

Next, read the following document that explains JSNI overlay types:

<http://code.google.com/p/google-web-toolkit/wiki/OverlayTypes>

I have used the overlay type concept in my OpenLayers wrapper code. It **greatly** simplifies the code over the old way of pairing a "native" class with each wrapper class.

3.8.3.2.3.2 OpenLayers 2D Map Client

OpenLayers is an open source JavaScript library for enabling feature-rich 2D geographic maps in a web browser. The OpenLayers web site is here:

<http://openlayers.org/>

3.8.3.2.3.2.1 Core OpenLayers Library

First off, you'll want to know where to find the reference guide for core OpenLayers classes. The easiest way is to just Google on the full class name. For example, the reference page for **OpenLayers.Layer.Vector** is here:

<http://dev.openlayers.org/docs/files/OpenLayers/Layer/Vector-js.html>

When I created the GWT wrapper classes for the OpenLayers classes that we needed, I spent a lot of time pulling up these reference pages and making sure that I was submitting the correct JavaScript object types and expecting the correct return types. JavaScript tends to just fail with little useful description of the problem being displayed. It is important to get the wrapper methods right the first time, and to manually test each new wrapper method and ensure that it works. Trust me when I say that the OpenLayers documentation is **not** 100% correct! If you don't test each method that you wrap to confirm that it works, you will pay for it later!

3.8.3.2.3.2.1.1 OpenLayers Distributions

The OpenLayers distribution can be obtained here:

<http://openlayers.org/download/>

The current version of JNDMS Portal is integrated with OpenLayers 2.7 distribution. However, be aware that it was necessary to apply several patches to the core distribution. OpenLayers 2.8 did not become available until June 22, 2009. In the interim, we had to make due with the limitations of release 2.7 and patch as required. In order to keep track of the patches, I put the entire OpenLayers 2.7 distribution code under Subversion source control in the JNDMS project repository. It can be found here:

```
/libs/OpenLayers/OpenLayers-2.7/
```

I have tried to be as descriptive as possible in my Subversion commit comments, so anyone who needs to maintain this code will know what OpenLayers defects have been patched.

3.8.3.2.3.2.2 **Building OpenLayers from Distribution Source**

Although the OpenLayers source code has been patched, this does nothing to update the assembled JavaScript library file. You will need to rebuild the library each time you modify/patch the core library source code. The instructions to do so are located here:

```
/libs/OpenLayers/OpenLayers-2.7/JNDMS README.txt
```

The instructions will describe how to create a regular version of the library, or an uncompressed version. The uncompressed version is human-readable and can be used for JavaScript debugging at runtime. The regular version is useless in both respects, but is compressed in order to execute faster. If you're not worried about optimizing for performance, such as during development, I highly recommend the uncompressed version. This will allow you to diagnose runtime exceptions in the OpenLayers JavaScript.

Since this is a GWT project, you also need to ensure that a reference to the JavaScript library is added to a GWT XML configuration file. I have already done this, but it is good to know the location of this configuration file:

```
/src/com/mdacorporation/gwt/openlayers/OpenLayers.gwt.xml
```

3.8.3.2.3.2.3 **Custom OpenLayers Code**

In order to implement the desired features for JNDMS Portal, a significant amount of custom extensions to the core OpenLayers library were created. Much of this was done by Peter Lynch prior to my joining the JNDMS team. I have cleaned up Peter's code and stored the scripts in the following location:

```
/war/js/map/openlayers/
```

I mentioned in a previous section that you should make yourself familiar with JSON (JavaScript Object Notation). Most of the JavaScript code in the core OpenLayers library source uses JSON notation. You will need to use it when extending core classes to implement your own, as you can see in Peter's existing code.

3.8.3.2.3.2.3.1 Composite Cluster Layer

The most significant piece of custom OpenLayers code is the “composite cluster” layer. When Peter first integrated the OpenLayers library with JNDMS, there were two requirements that could not be satisfied “out-of-the-box”.

1. Group together placemarks that are geographically close together and display them as a single “cluster” placemark
2. Provide ability to handle mouse click events on any placemark in any displayed layer (not just the top z-order layer)

The clustering problem could be solved by using the OpenLayers “cluster” strategy, and implementing some code to define the rules needed for JNDMS. The “handle clicks on any layer” was a missing feature in OpenLayers 2.7 (possibly implemented in OpenLayers 2.8). Peter chose to solve both of these issues at once.

The composite cluster layer is defined in this file:

```
/war/js/map/openlayers/mda-cluster_composite.js
```

Other supporting classes are defined in these files:

```
/war/js/map/openlayers/mda-composite_vector_layer_manager.js
```

```
/war/js/map/openlayers/jndms-cluster_composite_select_feature.js
```

```
/war/js/map/openlayers/jndms-cluster_style_calculator.js
```

```
/war/js/map/openlayers/mda-noop-renderer.js
```

What the code does is create a new layer that is a “composite” of one or more other data layers. The data layers are never directly rendered on the map, as they are supplied with a “no-op” renderer at creation time. The composite layer takes care of rendering the data from all of its data layers. Before this happens, each data item is run through the clustering logic to determine if one or more clusters should be created. Data items that are part of a cluster are not rendered. Instead, a circle is rendered to represent each cluster. Because everything that gets rendered is part of the composite layer, only that layer needs to handle mouse click events, thus solving issue #2 above.

3.8.3.2.3.2.3.2 Other Custom Classes

I will not describe the other custom classes here. Just open each JavaScript file and read the included comments to see what they are used for.

3.8.3.2.3.2.4 OpenLayers GWT Wrapper Classes

I have created GWT wrapper classes for the core OpenLayers JavaScript classes that we required for JNDMS Portal in this package:

```
com.mdacorporation.gwt.openlayers.client
```

GWT wrapper classes for OpenLayers JavaScript classes that were part of Peter's custom code extensions are defined in this package:

```
com.mdacorporation.gwt.openlayers.extensions
```

3.8.3.2.3.2.4.1 Code Patterns Used

If you have not read the articles that I suggested above in the GWT section, then this is your second warning that you should do so!

My OpenLayers wrapper classes mostly fall into two categories: those that are created with an "options" parameter, and those that are not. Most of the OpenLayers core library classes have constructors that accept an "options" parameter as the last parameter. Sadly, their documentation rarely mentions anything about what can be placed in the options parameter, so I'll explain.

With a few exceptions, any property that is listed on the OpenLayers documentation for a class can be supplied as a property in the "options" object at creation time. This includes properties that are inherited from a base class (which may explain why they did things this way).

To accommodate this, I have created a static inner class, as a container for options properties, with each class that supports options in its constructor. For example, if you open the `WMSLayer` class, you will see that it contains a static inner class named `WMSLayerOptions`. A `WMSLayerOptions` instance is required in order to create a new instance of `WMSLayer`. In this way, I can control what options are available to programmers who use the GWT wrapper classes. Options that I have tested and confirmed to work properly are included in the options class definition, while the rest are simply not included in the wrapper.

`WMSLayer` is also a good example of how I handle the wrapping to OpenLayers classes that are part of an inheritance hierarchy. Due to the limitations of GWT, you cannot create a new instance of a wrapped JavaScript object by using the Java `new` operator. Instead, I have created a static method called `newInstance()` for each of the wrapper classes. Per the GWT documentation, each class has a `protected` no-arg constructor defined. I have defined each class method as `final`, since the GWT documentation states that the framework cannot support method overriding. Wrapper classes that extend a base class can add new methods, but never override existing ones!

3.8.3.2.3.3 Google Earth Plugin 3D Map Client

The Google Earth Plugin is used to provide 3D map capabilities to a client browser. Since the KML schema originates from Google, it is fully supported by their Google Earth product.

The JNDMS team obtained a snapshot copy of GWT wrapper classes for Google Earth Plugin from Chris Mikkelsen. Chris was doing prototyping for another project team at the time, and was able to give us this in order to get a head start on our own integration. The wrapper classes are in this package:

```
com.mda.aif.frontrunner.client.widget
```

The level of effort to integrate with Google Earth Plugin was must lower than that for OpenLayers. The trade-off is less flexibility, less customizability, and higher cost (for a production license).

There is not much to say about the Google Earth Plugin. It supports any KML file that you can throw at it. It should support KMZ files as well, but I have had some problems getting it to accept them. This could be a configuration problem, and I have not spent much time looking into KMZ support.

One thing that you should be aware of is the license key. The free version of Google Earth Plugin (which is what JNDMS Portal currently uses) requires a license key on the client. Scott MacDonald created the current license keys, and it is my understanding that each key is tied to the IP address of the client machine. Even if I have that wrong, the important thing here is to know that a license key IS required to use Google Earth Plugin. I believe that they are free, but you have to create a Google account to obtain one. If a valid key is not present, then the plugin will not initialize.

3.8.3.3 Generic SQL

This section describes the Generic SQL system used to abstract and build SQL queries for the JNDMS Portal component. Starting with an explanation of the overall concept, the generic SQL implementation will be described, followed by a description of advanced generic SQL functionality.

3.8.3.3.1 Generic SQL Concept

The Generic SQL system is intended to provide an abstraction layer between the JNDMS Portal Java source code and the SQL queries used throughout the Java source. An abstraction layer allows for the creation of queries to be less error-prone and more consistent among different JNDMS Portal components, such as the map, the data grids, the topology applet and the detail pages.

A typical generic query defined in Java is composed of the following 3 parts:

- The *focus* indicates the database table on which to query
- The *filter* is a list of conditions for refining the query (Optional)
- The *sort* is a list of fields on which to sort the results (Optional)

In order to translate the abstract query into SQL, an SQL Builder class uses the following 2 types of configuration data stored in the database:

- The *Generic SQL Fields* indicate which columns to query for each *focus*
- The *Generic SQL Joins* indicate how to relate one *focus* to another

By defining fields and joins in the database, new fields and joins can be defined without having to rebuild the Java components.

3.8.3.3.2 Generic SQL Implementation

A basic SQL query is composed of the following 4 clauses: SELECT, FROM, WHERE and ORDER BY. The generic SQL system constructs each of these 4 clauses in turn by using the fields in an instance of a PageView object

(`com.mdacorporation.jndms.jui.client.data.PageView`).

3.8.3.3.2.1 SELECT

In order to construct the SELECT clause, the generic SQL system starts with a focus, stored in an instance of a PageView object. This focus is an enumeration corresponding to a database table. To determine the columns to include in the SELECT clause, the *GenericSQLFields* database table is queried. This database table defines a list of columns for each focus type. The *GenericSQLFields* table contains the following 5 columns:

- FOCUS: the database table name (e.g., 'Asset')
- SORT: the order number for the generic columns of a given focus
- FIELDSQL: the SQL used to construct the column. This can be the name of a column in the focus table, or it can be a subquery
- DISPLAY: the name of the column to be displayed to the user (e.g., in a data grid column header)
- FIELDID: the name to assign to the column within the generated SELECT clause (must be unique per focus)

A generic field can have metadata attached to it in the form of a suffix appended to its FIELDID column. This metadata is typically used to provide formatting information for generic columns being displayed in a data grid. The following suffixes are supported:

- HIDDEN: a field with this suffix is not meant to be displayed in a data grid. Most focus types have a PRIMARYKEY_HIDDEN field that is required for advanced generic SQL functionality, such as paging.
- LINK: a field with this suffix field provides a hyperlink for a corresponding field identified by the same FIELDID without the LINK suffix. Note that such hyperlinks are not used directly in the JNDMS Portal, but are instead converted into PageView instances in the `com.mdacorporation.jndms.jui.server.dbPageViewGenerator` class.
- SORT: a field with this suffix provides the means to sort a corresponding field identified by the same FIELDID without the SORT suffix. For example, if a user attempts to sort a generic query by the Asset Risk generic column, the generic SQL system will instead sort the generic query with the Asset Risk_SORT column. Fields that do not have corresponding SORT fields can still be used to sort.
- HIGHLIGHT: a field with this suffix provides a highlight colour to a corresponding field identified by the same FIELDID without the HIGHLIGHT suffix. This metadata type was fully supported in the JUI Portal, but is not fully supported in the JNDMS Portal.
- (deprecated) EXPAND: a field with this suffix indicates that the field must support tree expansion, such as in incident parent-child expansion. This field was used in the JUI Portal and is no longer used in the JNDMS Portal.

3.8.3.3.2.2 FROM

This is the simplest clause, composed of the database table name corresponding to the focus.

3.8.3.3.2.3 WHERE

The WHERE clause is optional, and is included if the *filter* defined in the PageView instance contains any *conditions*.

The *filter* is defined as an instance of the PageViewFilter class (`com.mdacorporation.jndms.jui.client.data.PageViewFilter`). This instance is called the page filter. It contains a list of conditions, each defined as an instance of the PageViewFilterCondition class (`com.mdacorporation.jndms.jui.client.data.PageViewFilterCondition`).

A typical condition contains the following values:

- Condition type: such as *EQUALS*, *BETWEEN*, *LIKE*, *IN*, etc.
- Table: the database table on which the condition exists. If this value is empty, it is assumed to be the focus table. If this table does not match the focus, a join is constructed, as described below.
- Field: the database column on which the condition applies (requires the Table value, above, to fully specify the database column)
- Value1: the value of the database field, which must be surrounded by single quotes if the database column is of VARCHAR or similar type. This value may be a subquery for conditions of condition type *EXISTS*, for example.

Some conditions, like *BETWEEN*, use a second value, named value2. By default, conditions are case sensitive, but a condition can be made case insensitive by setting the corresponding Boolean value in the PageViewFilterCondition instance.

If there are multiple conditions in the page filter, they are combined though an AND operation during construction of the WHERE clause. If a condition should instead combined though an OR operation to the condition preceding it, the corresponding Boolean can be set in the PageViewFilterCondition instance.

If a condition is on a database table other than the focus table, then the SQL generated for the condition is in the form of an EXISTS clause, containing a join between the focus table and the condition table. To determine the subquery required for the join, the *GenericSQLJoins* table is queried. The *GenericSQLJoins* table contains the following 3 columns:

- Focus: the focus table
- JoinTarget: the condition table
- SQLString: the SQL defining the join

If a requested *GenericSQLJoins* record does not exist for a focus and join target, then the generated query will be invalid.

There is a second instance of the PageViewFilter class, called the user filter, which is stored in the session and can be toggled on and off by the user, thus adding or removing its use from a PageView instance. The user filter is created using the filter definition dialog box in the JNDMS Portal, and is intended for restricting all data viewed in the JNDMS Portal to data relating to a specified operation, location or other entity. SQL construction from the user filter, if it is included in a PageView instance, is identical to SQL construction from the page filter.

3.8.3.3.2.4 ORDER BY

This optional clause is generated from a list of Strings defined in an instance of a PageView object. This list is automatically set to a default value during initialization of a PageView instance. See the `PageView.setSort(focus)` method for a list of the default sort values. Note that the values in the sort list must correspond to columns that are included in the SELECT clause. Otherwise, the generated query will be invalid. If the sort list is empty, this clause is omitted.

3.8.4 Subsystem Interfaces

The Visualization subsystem interfaces with the Data Warehouse subsystem and the DSS Subsystem.

3.8.5 User Interface / Portal Design

The following section identifies the user interface or presentation of the JNDMS within the Portal environment. It includes notes on what is to be displayed and some navigation information as well.

3.8.5.1 Global

This section discusses portions of the portal design or layout that is relevant to the entire portal, just specific views or tabs.

3.8.5.1.1 Portal Layout

The general layout of the portal is shown in Figure 3-12 and the names of these general areas will be used in the portal discussions.

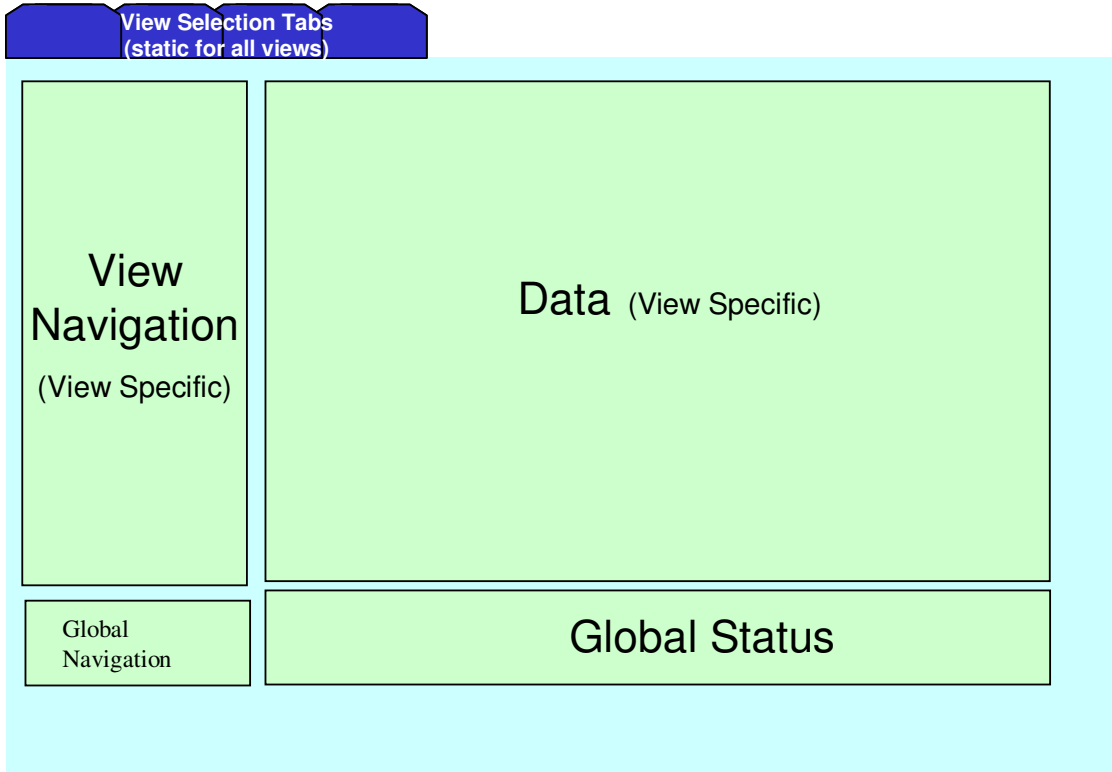


Figure 3-12: Portal Layout

The following guidelines apply to the portal layout:

- The navigation, and global status will have the title of the portlet set
- The data portlet should have the title set according to the current content. Some technical difficulties in applying the title to the data portal in which case the contents of the data portlet should clearly identify the content.

3.8.5.1.2 Login

The login is simply used to provide user credentials. The login for JNDMS is managed through the portal server (Liferay).



Figure 3-13: Login screen

3.8.5.1.3 Navigation

3.8.5.1.3.1 Navigation Overview and Guidelines

The navigation within JNDMS has three core views; the map (GIS, 2D and 3D), the details (HTML table views) and the visualization. Navigation should be consistent within any of these views such that links should keep the focus on the current view unless specifically identified as changing the current focus.

Throughout the portal three icons will be used to identify each of these views. See the icon section above.

The global navigation portlet can be used to manage switching between these focuses.

3.8.5.1.3.2 Tabs

The tabs are displayed the top of the user's view of the portal and provide a guide to access each of the primary views.

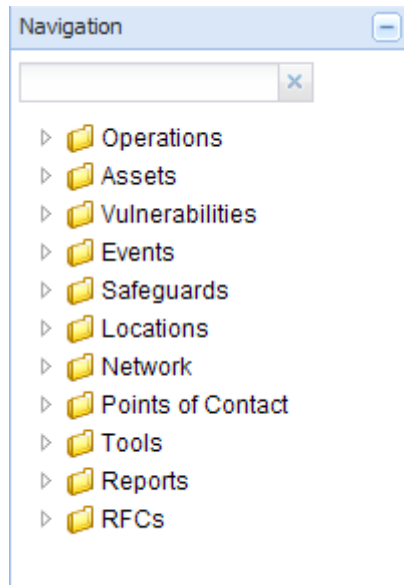


Guidelines for the tabs include:

- The current focus is maintained and as the new tab is selected the appropriate view will be shown.

3.8.5.1.3.3 Navigation Portlet

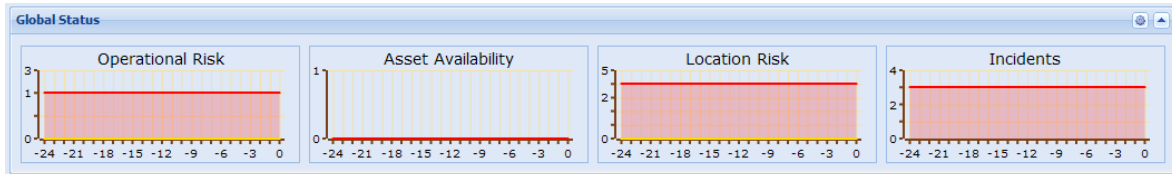
The navigation portlet is used to provide common navigation features in a tree view. Options chosen in this portlet will impact what is currently displayed in the data portlet.



- User may navigate through information trees and may expand or collapse branches of the tree
- Choosing content in the View Navigation Portlet will cause the Data Portlet to be updated
- Contents of the View Navigation Portlet itself will not be changed, except by expanding and collapsing tree branches. Only by selecting a tab can the user change the contents of the View Navigation Portlet
- Each tab will provide its own view of the navigation tree. See the sections specific to each tab for details.
- This navigation will maintain the global navigation guidelines such that the links will keep the global focus (map, details or visualization) unless explicitly stating that it will choose a specific focus.

3.8.5.1.4 Global Status

The global status view is shown at all times and is intended to give a quick of events, changes and the current risk analysis.



Global status guidelines include:

- Updates the summary every 10 seconds (configurable) to show the latest status of the COP
- Contents of the COP Summary Portlet can never be modified by an action resulting from a user selection

The information displayed in the global status includes the following:

- Four indicators
 - Operational risk. This will show a single bar that identifies all operations and it will be coloured according to the risks (red, green and yellow)
 - Location risk. This will show all locations and the associated risk.
 - Affected Assets. This will show all assets and how many currently have incidents logged against them.
 - Incidents. This will show the total number of active events and incidents. These should be shown as individual numbers, not a single bar.
- Below each of the four indicators there will be a graph showing the values over the last 24 hours.
 - The operation bar shows two line graphs. One for high risk, one for medium.
 - The asset graph is a single line graph showing impacted assets only.
 - The location graph will show two line graphs. One for high risk and one for medium risk.
 - The incident graph will show only incidents.
- Below the graphs we should show links to impacted operations and locations.
 - The initial version should show the text of operation and location names.

3.8.5.2 Maps

This section identifies the map views available.

3.8.5.2.1 General Map Guidelines

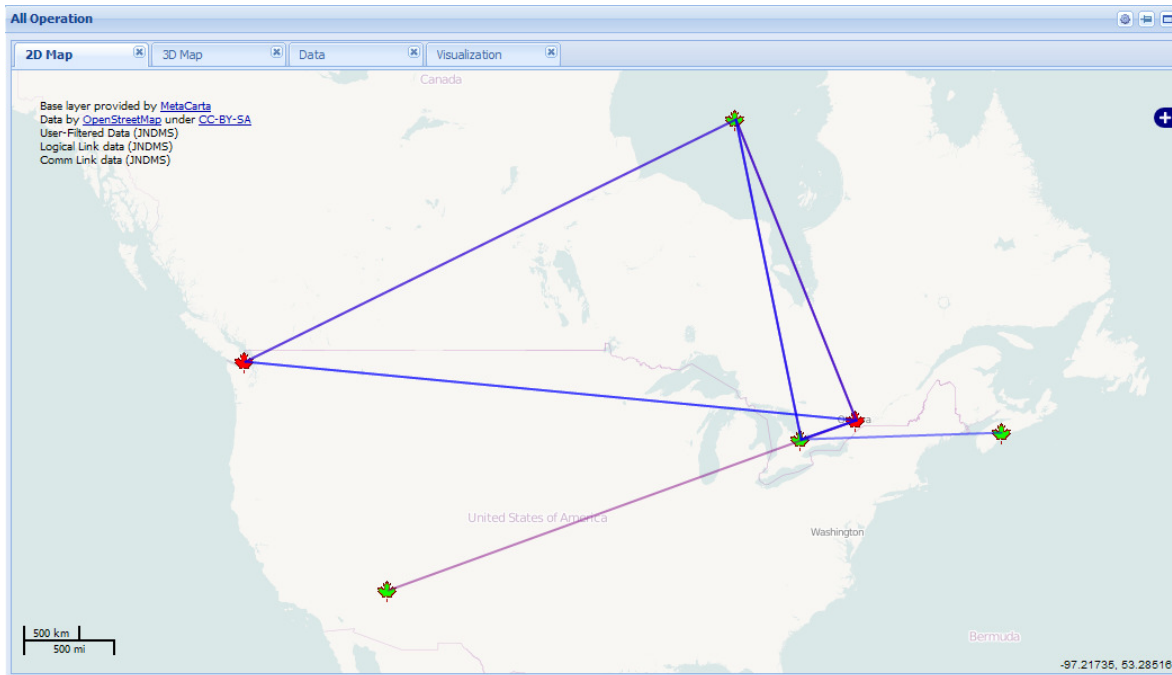
General guidelines:

- All content on maps such as colours, icons, lines, etc are dynamic and will refresh if the JNDMS data warehouse is modified.
- The map, when initially shown, will automatically zoom sufficiently to show all visible icons with enough borders to prevent icons from being split at the boundaries.

Icon guidelines for all maps

- Icons representing entities that are identified in the global highlight will have a highlight applied (highlights not currently implemented).
- Icons must identify recent changes in the information they are displaying. An alert decoration (see icon section above) should be applied whenever the status has changed within the last hour.
- Guidelines for drawing locations
 - Locations should account for both the central location as well as the bounding box if defined. If the bounding box would be sufficiently larger than the icon (50% guideline applied initially), then the full polygon should be drawn
 - Location polygons should be drawn as a transparency to allow for overlapping locations.

3.8.5.2.2 Map (Operation Summary View)



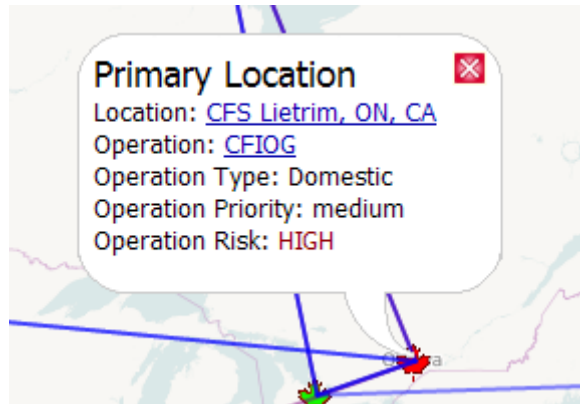
Content:

- Shows a map containing the minimum bounding area covering all JNDMS operations.
- The position of each icon is based on the location of the primary operational area.
- The title "Operations" should be clearly identified either in the title of the portlet (preferred) or within the map portlet.
- Communication links:
 - Comm links are an aggregate of all links between the operations. They represent communications identified between operations.
 - A link will be identified between operations if there are communication links between assets provided by (provisioned, see relationships in section above) an operation and assets provided by another operation.
 - Com links should be considered bidirectional for this display and only one link shown between operations.

- Dependent links:
 - Shows dependencies between operations. An operation will define its requirements through the opdetail records. This will identify the primary asset (host, machine, etc) with access to the service required. The links through the dependent/redundant tables will identify other dependant assets. If any of these assets are in another operation then an inter-operational dependency has been defined.
 - Colour should indicate:
 - Green (blue border). All direct requirements are currently met
 - Yellow (blue border). Degraded service (partial impact). The yellow indication cannot currently be displayed.
 - Red (blue border). Loss of capability.

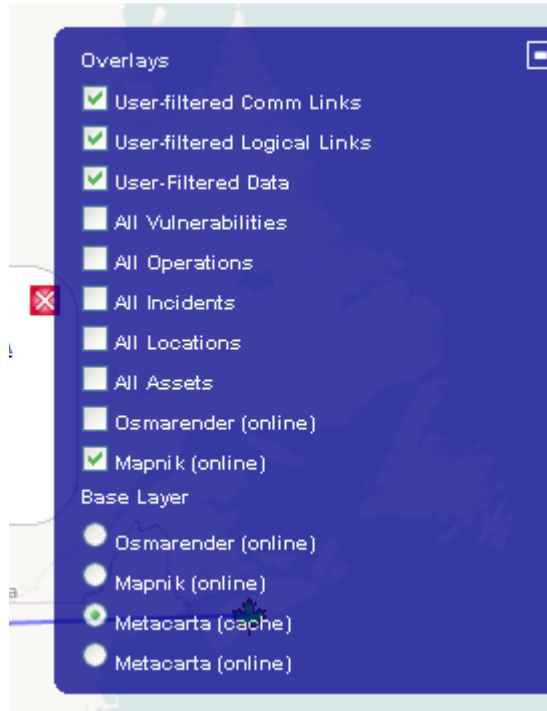
The following guidelines apply to the navigation within the maps:

- Popups on operations when mouse over occurs
 - Only one popup should show at one
 - The popup should have a cancel (red X) button to remove window



- The windows will show:
 - Operation name (hyper linked icons to operation details, map visualization or data)
 - Current risk
 - Number of locations (hyperlinked icons to location list filtered by operation, location map or visualization)
- Clicking on operation will show operation detail map

- Ability to show or hide based on common features:
 - Show/hide com links
 - Show/hide dependency links



3.8.5.2.3 Map (Operation details View)

The operation details view of the map shows locations within an operation. It will show one operation at a time.

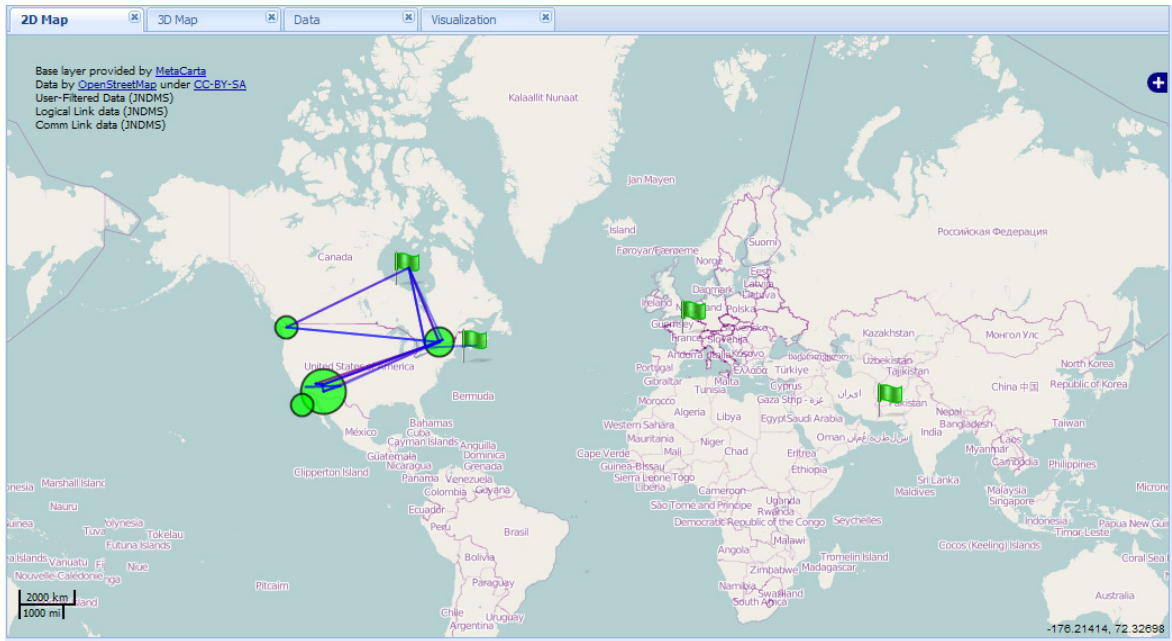
Content:

- Icons show locations within an operation (see location drawing guidelines for maps above)
- Colour of icons shows operational risk to current operation contributed by each location.
- Title bar (or map widget title) must identify that this shows operational risk for a specific operation.
- Preferred location display (note that cycle 2 implemented will show all locations that have assets assigned):
 - All top level locations should be shown (location without parent locations) if any assets are directly assigned to them.
 - For each top level location, drill down to the next level and determine if there would be overlap of icons if displayed. Keep drilling down as long as overlap does not occur. The resulting locations that don't overlap and have assets assigned to them should be shown.
 - If a location represents multiple locations have a link in the popup to expand or contract this location (contract this and all peers into parent)
- Links between icons indicate communications and requirements.
 - Comm links are shown if communications exist between assets in the given locations
 - Dependent links are shown if this operation requires an asset in the other location.

Links/Navigation:

- Clicking on icon shows asset summary view for given location
- Mouse overs show:
 - Operation name
 - Location name
 - Risk (contributed to current operation) at this location (not implemented in cycle 2)
 - Number of assets at location
 - Incidents at this location

3.8.5.2.4 Map (location summary view)



Content:

- Shows each area and status.
- Location display and navigation is similar to the operation details view except the colour of the icons and the links represent all operations, not just one.
- Incident locations should be shown as well as the operational locations
 - Be able to disable/hide the incidents.
 - Incident locations should follow the global location drawing guidelines.

Navigation:

- Poppers for icons and links same as for operation details, however reference all operations.
- Clicking on icon will show assets details for given location.

3.8.5.2.5 Map (zone/network view)

Content

- Show icon for each network zone. Position of a zone is based on locations with assets in that zone.

Navigation

- Mouseovers show zone id and link to zone details.
- Communication links show paths between zones through zone borders (cycle 3)
- Map (incidents and events)

Content

- Show individual incidents following global icon guidelines.
- No links or relationships between events are shown.

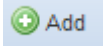
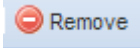
Navigation

- Mouseovers
 - Number of incidents
 - Impact summary
 - Operations impacted
 - Assets impacted
- Click will bring to incident details (icons showing details view or visualization)

3.8.5.3 Core Entity Views

3.8.5.3.1 Common Features

There are a number of common features that can be found in many of the core views. The following table identifies some common features:

Feature	Icon or View	Description
Add		This indicates the ability to add in the current view. It is generally found at the bottom of lists.
Delete		This indicates the ability to delete in the current view. It is generally found at the bottom of lists.
Pagination	Page 2 of 3 Previous Next	This is generally shown at the bottom of a list and indicates that there is more information available. 'Next' or 'Previous' are hyper links available if you can navigate forward or backward in this list.

3.8.5.3.2 List Assets

This view will show a list of assets. Each column will have a drop down box to filter the results. At the bottom of the list there will be controls to add entries, delete entries and an indication of how many pages of this list exist. The ability to at least navigate to the next page must be present.

Columns (selectable):

- Name
- Location
- Type
- Category
- Operation
- Zone
- Status
- Risk
- # Incidents
- Availability

3.8.5.3.3 Asset Detail Page

The asset details page will show the detailed contents known about an individual asset.

The following shows an example page showing the required fields and approximate layout. Each of the individual tables showing the relationships to the asset are generated from a generic query that is also used in other views. For example when showing the asset vulnerabilities, the same component and view is used to show the list of vulnerabilities. The view in this page however must be filtered based on this asset.

- General Information
 - Category / Type.
 - Status. This identifies if this is a new asset or part of an approved configuration.
 - Enabled. This identifies if this asset is enabled. An asset can be disabled if it is known to be in storage or transit.
 - Created/Modified. Time stamps for this asset.
 - IP Address. The IP address of this asset. This may be the default IP address for a host (if it has more than one IP address) or it may be the IP address of the network interface card (NIC).
 - Vendor/Product/Version. If this is a known product the information would be found here.
 - Importance. This is a rating identified by the JNDMS analysis based on how this asset is used.
 - Heart rate. Possible identification of how often this asset reports or is scanned (generally for availability).
 - Latitude/Longitude. Position information.
 - Network host. If this is a peripheral or network card this will identify the associated host.
 - Active incidents. This shows active incidents associated with this asset.
 - Risk override. This will identify if risk override is in effect. The user can override the calculated value of risk if they feel it is not appropriate. At any point the risk override can be disabled and the automatic risk value would be used.
 - Risk. This is the current risk value. This could be either a user entered value if the risk override is enabled or the DSS calculated risk score.
 - Availability. This is the current availability of this asset. This could be either up, down or degraded.
 - Asset Count. This asset could represent more than one asset. This would show how many identical assets this represents.
 - Analysis. This provides a textual description of the risk analysis.

- Asset vulnerabilities. This shows all vulnerabilities associated with this asset.
- Vulnerability ports. This shows the ports that have vulnerabilities current associated with them. This information generally depends on a vulnerability scanner providing port details.
- Safeguards implemented by Asset. This identifies the case where this asset is a safeguard.
- Assets protected by this asset. If this asset is a safeguard this will show the assets being protected.
- Assets that protect this asset. This will show safeguard assets that are protecting this asset.
- Asset that protect this asset through the zone. This will show perimeter safeguards that may be protecting this asset. The protection of perimeter safeguards depends on the path taken through the network.
- Asset availability. This will identify the availability of this asset.
- Operations. This identifies the associated operations.
- Asset locations. This identifies the associated locations.
- Zone containing asset. This identifies the zone or zones that contain this asset. An asset may exist in more than one zone if it is multi-homed.
- Incidents involving asset.
- Asset required by this asset
- Assets requiring this asset
- Redundant assets
- Communication links
- Points of contact
- RFCs
- Product
- Change log

3.8.5.3.4 Vulnerability Definition Details Page

- General info
 - ID
 - Category
 - Description
 - Created/Modified
 - Source
 - Type
 - Security Level
 - Status
 - Number of Products
- Exploits
- Products
- Assets
- Safeguards
- Incidents
- RFCs
- Change Log

3.8.5.3.5 List Safeguards

Safeguards

- ID. This is a unique identifier.
- Type. This is the type of the safeguard such as firewall or patch.
- Description. This is a textual description of the safeguard.
- Data Source. This is the source that reported this safeguard.
- Efficiency C/I/A. This identifies how effective this safeguard is in protecting Confidentiality, Integrity and Availability. This allows for cases where safeguards are only partially effective.
- Policy ID. This gives an link to potential policies. This is for reference only.
- SensorSigID. This gives a potential link to signatures that are tracked.
- Modified. The last modified date.

3.8.5.3.6 Safeguard Details Page

- General info
 - ID
 - Data Source
 - Efficiency [C/I/A]
 - Policy ID
 - Sensor Sig ID
 - Created/Modified.
 - Description
 - Type
- Assets implementing
- Assets protected by
- Safeguarded vulnerabilities
- Zones bordered
- Zone rules
- RFCs

3.8.5.3.7 List Operations (operation summary)

- Operations. This will be a list of operations
 - Name. The name of the operation.
 - Type. The type of the operation.
 - Priority. The priority (between operations) given to this operation.
 - # Sites. Sites (locations) associated with this operation.
 - # Incidents. Incidents associated with this operation.
 - Risk. The current risk score for the operation.

3.8.5.3.8 Operation Details Page

- General Information.
 - Name of the operation.
 - Type of the operation, such as domestic or deployed.
 - Priority. This is a value to compare different operations.
 - Command officer.
 - Risk. This is based on the combined risk associated with any asset required by this operation.
 - Analysis. This provides a summary of the risk analysis done for this operation.
- Dependencies. This provides a tree view of the operational dependencies.
- Operational Assets. This provides a flat list of all of the assets that this operation depends on. This view of the assets shows the relationship to this operation.
- Assets. This shows the dependent assets with the columns found in the generic asset lists.
- Vulnerabilities. This shows all vulnerabilities associated with this operation.
- Safeguards. This shows all safeguards associated with this operation.
- Locations. This shows all locations associated with this operation.
- Zones. This shows all zones associated with this operation.
- Incidents. This shows all incidents associated with this operation.
- Operational Events. This shows operational events associated with this operation. An operational event gives windows of time for the expected activity of the operation. Most operation will have a default event that spans the life time of the operation.
- Units. This lists who (what units) comprise this operation.
- Points of Contact. This identifies points of contact for this operation.

3.8.5.3.9 List Locations (location summary)

- Name. Name of the location.
- Description. Textual description.
- Latitude / Longitude. Position information.

3.8.5.3.10 Location Details Page

- General Info
 - Name
 - Description
 - Building / Floor / Room / Rack
 - Street / City/Province/Status/Country/Postal Code
 - Radius
 - Notes
- Assets
- Vulnerabilities
- Vulnerability instances
- Operations
- Safeguards deployed
- Zones
- incidents

3.8.5.3.11 List Zones (zone summary)

- Zone. This is the network zone
- Risk. Risk associated with this network. Network risk is based on the assets within the zone.
- Probability of Attack. This is a computed value based on the events and vulnerabilities within the zone.
- #Subnets. The number of subnets that make up this network.
- #Safeguards. The number of safeguards within the zone.

3.8.5.3.12 Zone Details Page

- General info
 - Name
 - Description
 - Zone ID
 - Probability of Attack
 - Latent probability of attack
 - Risk
 - Created / Modified
 - Notes
- Assets protecting zone
- Assets in zone
- Vulnerabilities
- Safeguards
- Operations affected
- Locations affected
- Adjacent zones
- Incidents
- RFCs
- Zone rules
- Subnets
- Points of Contact

3.8.5.3.13 List Events

- ID. This is a unique identifier for the event.
- Root Event. This is a flag to identify if this event is a root event. In JNDMS a single event may be related to other events in a cause / effect relationships. The event that is the cause is the parent of those that are the effects.
- Type. This is the type of event, such as compromise or policy violation.
- Status. This is the status of the event such as active, resolved or mitigated.
- Location. This is the location associated with the impact of the event, if any.
- Asset. This is the asset associated with the event.
- Created. The date this event was created.
- SP* (DSS Priority). This is a priority calculated by the analysis to help identify events causing issues.
- Severity. This is the severity of the event based on the importance of the assets that were impacted.

3.8.5.3.14 Event Details Page

- General Info
 - Description
 - Incident ID
 - Created/Modified
 - Event time
 - Is Incident flag
 - Type
 - Disposition
 - Confidence
 - Status
 - Parent Incident
 - Notes
 - Alert
 - System Events
 - Operational dependency value
 - Source priority
 - Analysis priority
 - Security level
 - Logs
 - Formatted reports
 - Location
 - Ticket ID
 - Ticket Status
 - Severity
 - Data Source

- Incident Sensor Path. This identifies, when available, the IP address, the asset name, the receive time, the completion time, the event type, the severity and the priority of this event. This chart identifies each of these values with respect to the source, the sensor, the target and the JNDMS analysis.

- Impact
 - Success probability
 - Is Incident
 - Vulnerability ID Type
 - Affected Asset
 - Priority
 - Alert
 - Environmental Damage
 - Environmental Damage Value
 - Environmental Damage Base Score
 - Impact (C/I/A)
 - Severity
- Child Incidents (tree view)
- Assets
- Affected Assets
- Source of the incident
- Location of the sensor
- Affected Operations
- Associated Vulnerabilities
- Correlated Incidents
- Correlation Details
- Zones
- SOPs
- History

3.8.5.4 Report Pages

Reports are generated with a tool called Jasper Reports. This tool can be used to create new reports and publish the template to JNDMS using the 'Add Report' link.

3.8.5.4.1 Select Report

Select a report type:

Select a report format:

3.8.5.4.2 Add New Report

New reports are created with Jasper Reports based on the JNDMS data model. These reports can be published using this form.

Add New Report

Report Name:

Report Definition File:

3.8.5.5 Secondary Views

The secondary views indicate additional views that have been identified as required but may actually be relationships between the core entities. For example the "vulnerability instances" view (particular vulnerabilities on a particular asset) could be shown to display relationship between the assets and related vulnerabilities. This section will show views designed for these secondary entities.

3.8.5.5.1 Show Vulnerability Instances

- ID. This is the ID of the vulnerability definition.
- Source. This identifies the source that identified the vulnerability. This could be a reporting vulnerability scanner or it could have been an analysis done by JNDMS.
- Status. This will identify if the vulnerability is 'new' or 'mitigated'.
- Modified. The last time an update was received for this instance.
- Asset Name. The name of the asset that is vulnerable.
- Asset Category. The category of the asset that is vulnerable.

3.8.5.5.2 Exploits

Exploits. Exploits are components of vulnerabilities in the JNDMS model. These identify one or more methods that the vulnerability could be exploited. Many sources don't separate the exploit from the vulnerability so for these cases a 'default' exploit is made for each vulnerability.

- VulnID. This is the ID of the associated vulnerability.
- ID. An identifier that is unique for the given associated vulnerability. Many exploits will have the ID 'default' to ensure that at least one exploit is tracked.
- Description. This is the text description for the exploit.
- Availability. This identifies if the exploit is unproven or widely available (based on CVSS scoring).
- Date. The date this exploit was identified.
- Access Vector. This identifies if this requires local access or if it can spread over the network.
- Authentication. This identifies if authentication is required for this exploit to be effective.
- Popularity. How popular this exploit is.
- CVSS Score. The CVSS score (from CVE entries).

3.8.6 Visualization Applet

3.8.6.1 Visualization overview

This is a brief discussion on the visualization applet for JNDMS. The relationships discussed here are also relevant to the other views (map and table).

The proposed applet would show the relationships to an entity that has the current focus. Although JGraph doesn't have the direct concept of a center object in its layouts the idea would be to build the graph from this entity outwards, let JGraph layout the graph and identify (possibly a circle around the entity) the center object. Figure 3-14 shows a visualization of all operations, drilling down to opareas, drilling down further to opassets.

There are several features within JGraph that would add to the visualization. The first is the automatic layout. We could allow the user to select from the layout algorithms (see Figure 3-14).

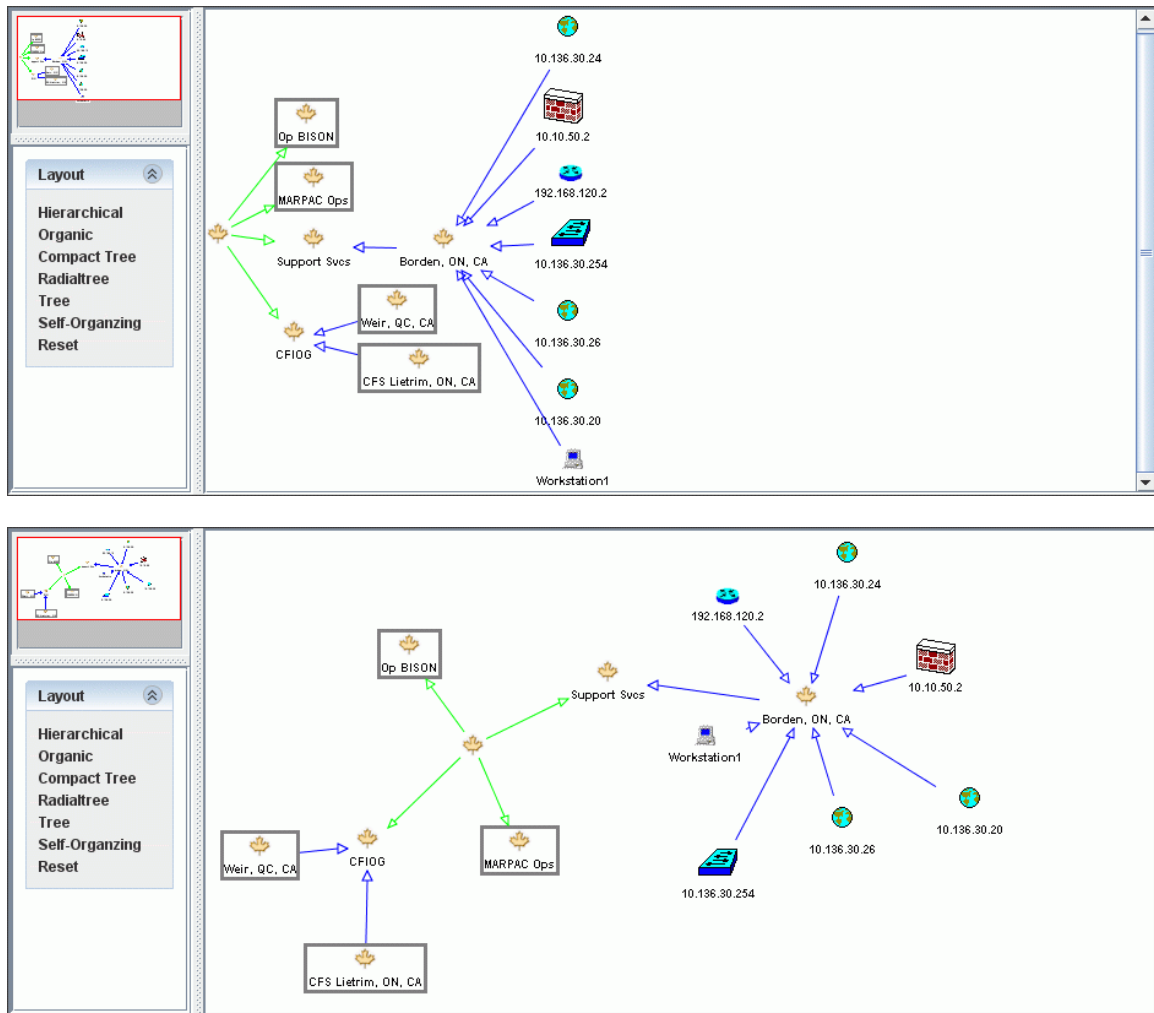


Figure 3-14: Layout options for JGraph: “Compact Tree” (top) and “Organic” (bottom)

The next feature would be the view port widget combined with the zoom feature (see Figure 3-15, with view port widget visible in upper left). This would allow the user to zoom in and out while still maintaining the context of what they are looking at. Only when zoomed in would you have any text on the icons. See Figure 3-16 for a zoomed-out visualization.

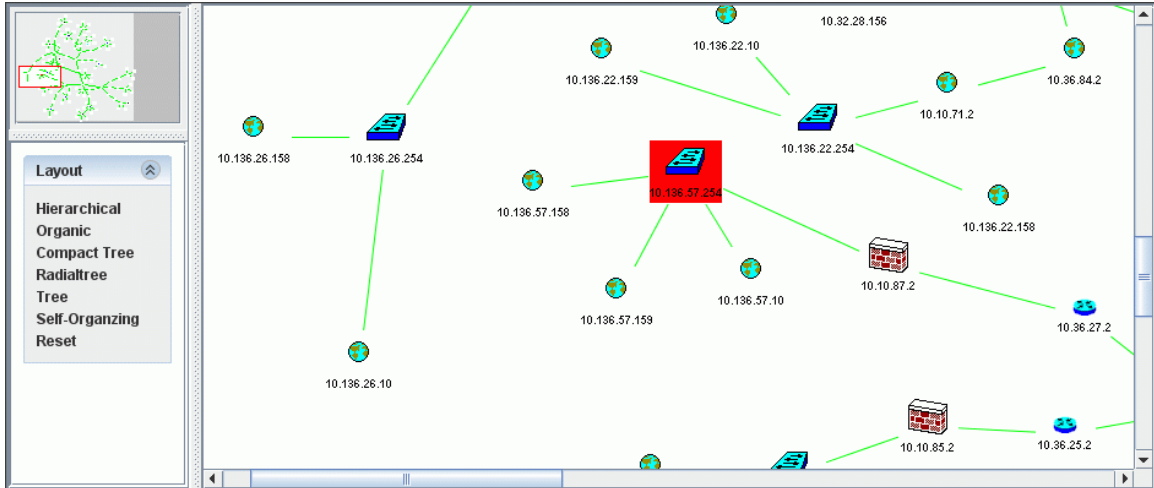


Figure 3-15: JGraph view port widget, including highlighted asset at risk

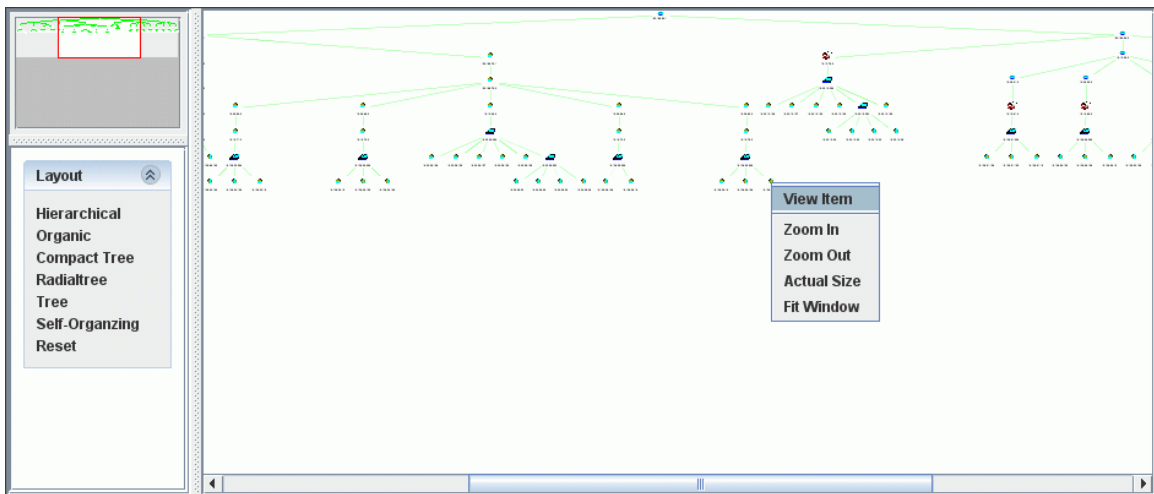


Figure 3-16: Zoomed-out visualization with zoom context menu, including link to detail page

Some of the interaction with the applet would include:

- The context menu (right click) would be used to selectively expand or collapse the views.
- A click on the node would expand or collapse the node showing or hiding its relationships.
- A double click would make that node the new focus.
- The global filters and highlights would be active. Link on the nodes would allow quick editing of the filters and highlights.

The above rules should be able to create most of what we would need. For example if the user is interested in a network topology type map they would only enable the 'asset' entity and only show the 'link' attributes. Assets may need to be treated as three separate entities such as hardware assets, software assets and capability assets.

3.8.6.2 Architecture

There would be two components to this visualization, the client side applet and the server side components. The server side would be responsible for all database access and communication with the client side applet. The work currently being done with the dynamic building of queries, global filters and highlights would form the basis of the server side components for the visualizations as well as the table views.

The important relationships for the primary entities would have to be shown in the detail pages for each entity type. This same information and the same relationships should be shown in the visualization. The visualization, however, would be able to show several links away.

The applets would have to have a connection to the backend server to make queries and to react to updates from the system. The same components can have an adaptor class to provide the data in the formats required for each of the clients. For example the tables may request HTML for the entity, however the visualization applet may request XML or another format, and will also likely request multiple layers at once. Most of the logic dealing with the queries and the relationships should be addressed server side so that reuse between the various JNDMS components can be maintained.

3.9 Situational Awareness Data Sharing

The SA data sharing subsystem is responsible for both inputs to the JNDMS, as well as data export from the JNDMS to other SA systems. These other systems may include peer JNDMS, a JNDMS on a different security level or other coalition systems.

The components developed for this subsystem makes considerable use of the data transformation services and the input and query services of the JNDMS.

The JNDMS can be configured to export data or share with a peer JNDMS. Each event is pre-processed, and then can be sent to an external system. The sharing at an event level allows analysis on each system to work independently with different views if the network environment.

The data sharing component is implemented as part of the JSS. The JSS will perform initial validation of the incoming events then check it to see if the given event should be shared (see section 3.6.3.1 for details on event stages).

3.9.1 Setting up data sharing

The data sharing, as part of the JSS, is configured within the web service properties (`jss.properties`). The format of this file is a standard Java property file. The following general properties can be set:

- `Jss.share.enable`: This can be either true or false. Sharing will only be active when this property is true.
- `Jss.share.allow`: This is a list of JNDMS global IDs that are explicitly allowed to connect. This is used in conjunction with the `jss.share.relay` settings.
- `Jss.share.deny`: This is a list of JNDMS global IDs that are explicitly disallowed from connecting.
- `Jss.share.relay.sensors`: This property has two parts, an enable flag and a default connection policy. This entry sets the ability of sensor data to be relayed through this JNDMS instance on to other JNDMS peers.

The enable flag can be enabled or disabled. The default connection policy can be either allow or deny. If the default policy is to allow, then all JNDMS peers will be allowed to connect except those explicitly listed in the `jss.share.deny` list. If the default policy is to deny, then only JNDMS peers listed in the allow list will be permitted.

- `Jss.share.relay.jndms`: This property is configured with the same parameters as `jss.share.relay.sensors`, however this refers to the policy of forwarding events that have been received from another peer JNDMS instance.

In addition to the above settings the data sharing must be configured with one or more data sources and one or more forwarding rules to these sources.

The format of a source line is as follows:

```
Jss.share.source.[id]=[jndms id], [source address]
```

Where:

- Id: This is an identifier to be used within the configuration file to refer to this source. The sources should be incrementally numbered starting with 0.
- Jndms id: This is the globally unique identifier used by the remote JNDMS instance.
- Source address: This is the URL of the remote JNDMS instance. This must either be the URL of the remote JSS web service or the URL of a directly used to propagate the events.

The format of a forwarding line is as follows:

```
Jss.share.forward.[id]=[source], [destination], [filters]
```

Where:

- Id: This is an identifier for the forwarding rule. Each rule must have a unique identifier.
- Source: This is the ID of one of the sources previously configured that refers to the source of the event. This can also be 'any' if all JNDMS peers should be considered.
- Destination: This is where to send the event and refers to either 'any' or one of the previously configured sources.
- Filters: This can list one or more filters separated by a '|'. Each filter has the format of the name of the filter and the filter parameters such as name=parameters. The following types and options are supported:
 - Type filter: This will list type;sub type pairs of events to forward.
 - Source filter: This will forward events from the given source

3.9.2 Sharing example configuration

The following configuration will set up JNDMS to share incoming events with two additional peers and to allow incoming events from JNDMS peers.

```
# Sharing configuration
jss.share.enable=true

# sensor and system (jndms) relay settings true/false,deny/allow
jss.share.relay.sensors=true,allow
jss.share.relay.jndms=true,allow

# source list
jss.share.source.0=2,http://192.168.0.4:8080/JSS/services/JNDMSPort
jss.share.source.1=3,file:/c:/jndms/eventQueue_topsecret/

# forwarding rules
# format: source,destination,filters (separated by |)
jss.share.forward.0=any,0,type=sim;exploit
jss.share.forward.2=any,3,none
```

The first peer configured represents a peer system that can be contacted directly. The second represents the case where the file system is used to share events. This may be used in cases where network shares provide the communication mechanism or where additional tools such as forwarders (one way data diodes) are in use.

The first forwarding rule will forward all events from any source of type “sim;exploit” to the JNDMS peer identified by ID ‘0’. The second rule will send all incoming events to the given system, in this case through the file system.

A JNDMS instance that is expecting incoming events through the file system can use the JSS Client to monitor the directory and any events that are available will be picked up and forwarded to the configured JNDMS web service. The following is the command line that can be used to scan directory ‘x:’ for incoming events:

```
java -jar jss_client.jar endpoint=[url of jndms] op=scan_dir dir=x:/
```

Appendix A Acronyms

ACL	Access Control List
ADT	Advantage Data Transformer
AFCCIS	The Canadian Air Force Command and Control Information System
AH	Authentication Header
AIS	Application Integration Server
AJAX	Asynchronous JavaScript and XML
AM	Asset Management
API	Application Program Interface
ARP	Address Resolution Protocol
ARS	Action Request System
ASCII	American Standard Code for Information Interchange
B2B	Business-to-Business
B2C	Business-to-Consumer
B2E	Business-to-Employee
BIOS	Basic Input/Output System
BPS	Boundary Protection System
BRE	Business Rules Expert
C2	Command and Control
C2IEDM	Command and Control Information Exchange Data Model
C4ISR	Command, Control, Communications, Computers, Intelligence, Surveillance, and Reconnaissance
CA	Computer Associates
CANUS	Canadian and US
CAPI	Cryptographic Application Programming Interface
CDRL	Contract Data Requirements List
CFNOC	Canadian Forces Network Operations Centre
CIA	Confidentiality, Integrity and Availability
CIK	Crypto Ignition Key
CIRT	Computer Incident Response Team

CMDB	Configuration Management Database
CND	Computer Network Defence
CNES	Canadian Network Encryption System
CO	Commanding Officer
CONOPS	Concept of Operations
CoS	Class of Service
COTS	Commercial Off The Shelf
CSE	Communications Security Establishment
CVE	Common Vulnerability Exposures
CVSS	Common Vulnerability Scoring System
DDE	Deployable DWAN Equipment
DEMS	Defence Electronic Mail System
DHCP	Dynamic Host Configuration Protocol
DID	Data Item Description
DMF	Device Modeling Framework
DMFD	Device Modeling Framework Definition
DND	Department of National Defence
DRDC	Defence R&D Canada
DREnet	Defence Research Establishment Network
DSS	Decision Support System
DVPNI	Defence Virtual Privet Network Infrastructure
DW	Data Warehouse
EAL	Evaluation Assurance Level. These levels are defined by the Common Criteria guidelines.
EIM	Enterprise Infrastructure Management
EKMS	Electronic Key Management System
ESP	Encapsulating Security Payload
ETL	Extract, Transform, Load
eT VM	eTrust Vulnerability Manager
FTP	File Transfer Protocol
GIS	Geographic Information System
GUI	Graphical User Interface
HIDS	Host Intrusion Detection System
HIPS	Host Intrusion Prevention System
HR	Human Resources

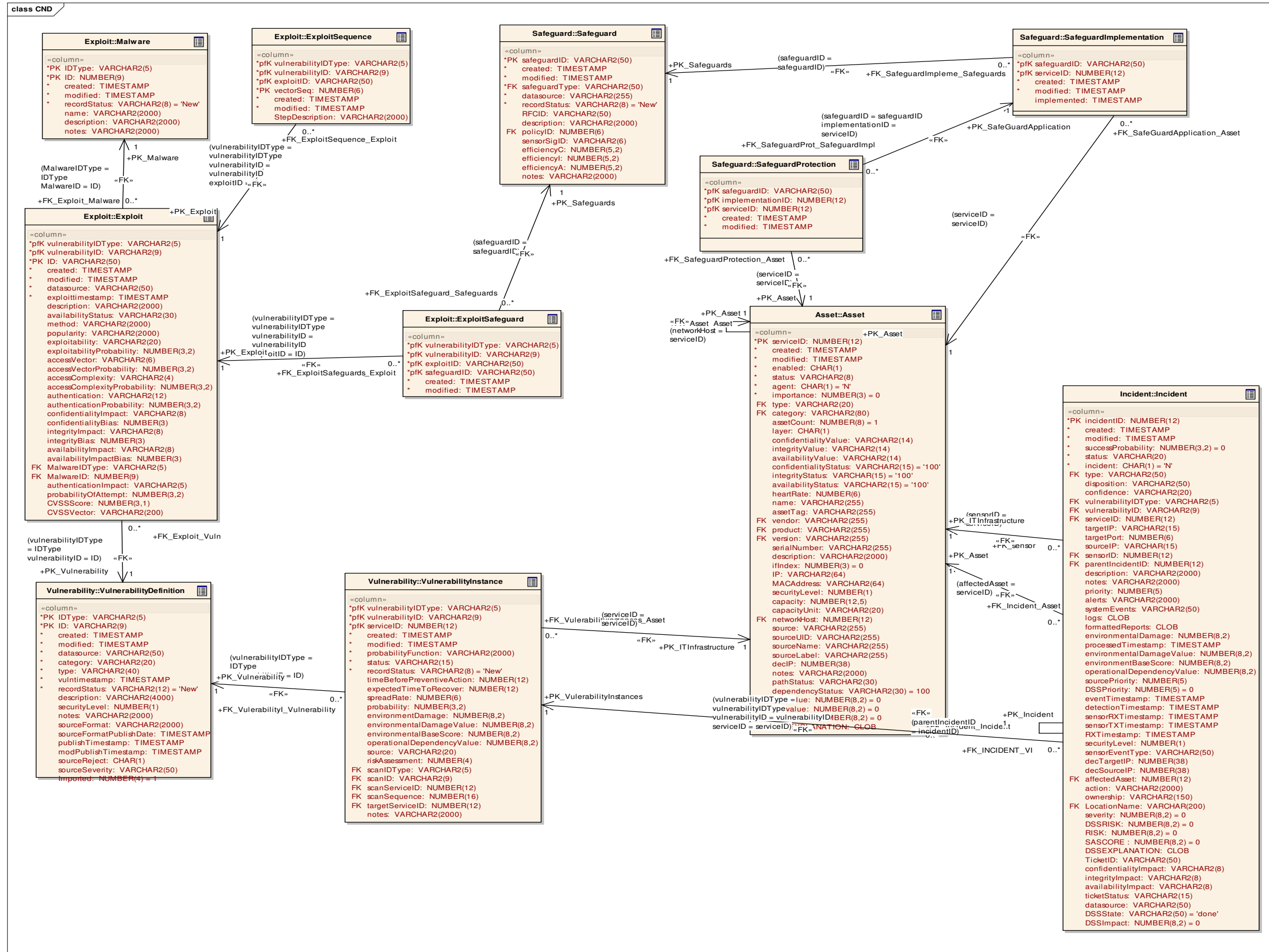
HTML	Hypertext Markup Language
HTTP	Hyper Text Transfer Protocol
I&A	Identification and Authentication
IAT	Impact Assessment Tool
IATF	Information Assurance Technical Framework
ICMP	Internet Control Message Protocol
ID	Identification
IDE	Integrated Development Environment
IDS	Intrusion Detection Systems
INE	In-line Network Encryptor
IP	Internet Protocol
IPSec	Internet Protocol Security
ISM	Intellitactics Security Manager
ISP	Internet Service Provider
IT	Information Technology
ITI	Information Technology Infrastructure
J2EE	Java 2 Enterprise Edition
JDBC	Java Database Connectivity
JDW	JNDMS Data Warehouse
JNDMS	Joint Network Defence and Management System
JSR	Java Specification Request
JSS	JNDMS Services
JUI	JNDMS User Interface
KMI	Key Management Infrastructure
LMP	Link Management Protocol
MARLANT	Maritime Forces Atlantic
MARPAC	Maritime Forces Pacific
MCOIN	Maritime Command Operation Information Network
MDB	Management Database. This refers to the datastore used by the CA products.
MDF	Main Distribution Frame
MOM	Microsoft Operations Manager
MTTRS	Mean Time To Restore Service
MUX	Multiplexer

NASL	Nessus Attack Scripting Language
NATO	North Atlantic Treaty Organization
NDHQ	National Defence Headquarters
NIAC	National Infrastructure Advisory Council
NIO	Network Information Operations
NIST	National Institute of Standards and Technology
NSM	Network Systems Management. This is part of the Unicenter product line.
NTSM	National Telecommunication Management System
NVD	National Vulnerability Database
ODB	Operations Database
ODBC	Open Database Connectivity
OOB	Out Of Band
OODA	Observe, Orient, Decide, Act
OpenGIS	Open Geodata Interoperability Specification
OSVDB	Open Source Vulnerability Database
PBX	Private Branch Exchange (private telephone switchboard)
PKI	Public Key Infrastructure
POC	Point of Contact
PWGSC	Public Works and Government Service Canada
QoS	Quality of Service
R&D	Research and Development
RDBMS	Relational Database Management System
RDEP	Remote Data Exchange Protocol
RFC	Request For Comments (Internet Standards documents)
RFP	Request for Proposal
RSS	Real Simple Syndication
SA	Situational Awareness
SCC	Security Command Centre
SCEM	Secure Common Email
SCI	Special Compartmented Information
SCP	Secure CoPy
SDA	Service Delivery Area

SDNS	Secure Data Network System
SDP	Service Delivery Point
SDW	Security Data Warehouse
SIM	Security Information Management
SIP	Service Interface Point
SML	Strength of Mechanisms Level
SMS	Systems Management Server
SMTP	Simple Mail Transfer Protocol
SNI	Secure Network Infrastructure
SNMP	Simple Network Management Protocol
SOAP	Simple Object Access Protocol
SRA	Secure Remote Access
SSH	Secure Socket Shell
SSL	Secure Sockets Layer
TBD	To Be Determined
TCP	Transmission Control Protocol
TD	Technology Demonstrator
TDP	Technology Demonstration Project
TT	Trouble Ticket
TTP	Trusted Third Party
UDP	User Datagram Protocol
UPS	Uninterrupted Power Supply
VA	Vulnerability Assessment
VPN	Virtual Private Network
WAN	Wide Area Network
WAR	Web Application Archive
WGS 84	World Geodetic System 1984
WSDP	Web Services Developer Pack
XML	eXtensible Markup Language
XSLT	extensible Stylesheet Language Transformations

Appendix B CND Data Model

The following diagram provides a detailed view of the key Computer Network Defence (CND) relationships within the JNDMS data model. This view includes the tables, attributes and major relationships.



Appendix C Sample Safeguard Data Inputs

Safeguard report

```
table=zone
headers=ZONEID,LABEL,CREATED,MODIFIED,DESCRIPTION,PROBABILITYOFATTACK,PROBABILITYOFATTACKLATENT,OPRISKC,OPRISKI,OPRISKA
1,Internet,11/21/2006 3:54:35.102645 PM,11/21/2006 3:54:35.102645 PM,The internet zone.,0.9,0.9,0,0,0
2,dnd.wan,11/21/2006 3:54:35.223496 PM,11/21/2006 3:54:35.223496 PM,From the internet firewall to each site firewall.,0.1,0.1,0,0,0
3,dnd.site.ottawa.internal,11/21/2006 3:54:35.236716 PM,11/21/2006 3:54:35.236716 PM,Ottawa servers and workstations.,0.1,0.1,0,0,0
4,dnd.site.ottawa.dmz,11/21/2006 3:54:35.248686 PM,11/21/2006 3:54:35.248686 PM,Ottawa DMZ.,0.2,0.2,0,0,0
5,dnd.site.corporate.internal,11/21/2006 3:54:35.257476 PM,11/21/2006 3:54:35.257476 PM,Corporate servers and workstations.,0.1,0.1,0,0,0
6,dnd.site.corporate.dmz,11/21/2006 3:54:35.265378 PM,11/21/2006 3:54:35.265378 PM,Corporate DMZ,0.2,0.2,0,0,0
7,dnd.site.halifax.internal,11/21/2006 3:54:35.278831 PM,11/21/2006 3:54:35.278831 PM,Halifax servers and workstations.,0.1,0.1,0,0,0
8,dnd.site.halifax.dmz,11/21/2006 3:54:35.301934 PM,11/21/2006 3:54:35.301934 PM,Halifax DMZ,0.2,0.2,0,0,0
```

zoneborder

```
# SELECT z.*, a.ip, a.name FROM ZONEBORDER z, ASSET a WHERE z.serviceid=a.serviceid;
```

```
table=zoneborder
headers=SERVICEID,ZONEID,CREATED,MODIFIED,SAFEGUARDID,SAFEIMPLSERVICEID,IP,NAME
67,2,11/18/2006 11:12:09.062168 AM,11/18/2006 11:12:09.062168 AM,,,131.136.51.2,fw-Halifax-131.136.51.2
68,7,11/18/2006 11:12:09.076296 AM,11/18/2006 11:12:09.076296 AM,,,131.136.52.1,fw-Halifax-131.136.52.0/24
69,8,11/18/2006 11:12:09.079212 AM,11/18/2006 11:12:09.079212 AM,,,131.136.50.1,fw-Halifax-131.136.50.0/24
15,1,11/18/2006 11:12:09.049196 AM,11/18/2006 11:12:09.049196 AM,,,131.132.1.2,fw-Internet Firewall-131.132.1.2
16,2,11/18/2006 11:12:09.052645 AM,11/18/2006 11:12:09.052645 AM,,,192.168.10.1,fw-Internet Firewall-192.168.10.0/24
50,2,11/18/2006 11:12:09.055708 AM,11/18/2006 11:12:09.055708 AM,,,131.136.21.2,fw-Ottawa-131.136.21.2
51,3,11/18/2006 11:12:09.065449 AM,11/18/2006 11:12:09.065449 AM,,,131.136.22.1,fw-Ottawa-131.136.22.0/24
52,4,11/18/2006 11:12:09.068234 AM,11/18/2006 11:12:09.068234 AM,,,131.136.20.1,fw-Ottawa-131.136.20.0/24
59,2,11/18/2006 11:12:09.058728 AM,11/18/2006 11:12:09.058728 AM,,,131.132.11.2,fw-Corp-131.132.11.2
60,5,11/18/2006 11:12:09.070929 AM,11/18/2006 11:12:09.070929 AM,,,131.132.12.1,fw-Corp-131.132.12.0/24
61,6,11/18/2006 11:12:09.073589 AM,11/18/2006 11:12:09.073589 AM,,,131.132.10.1,fw-Corp-131.132.10.0/24
```

DN0678
Issue 4/2: 06 February 2012

table=zonesubnet
headers=ZONEID,ZONESUBNETID,SUBNETID,IP,CREATED,MODIFIED,STARTIP,ENDIP
2,1,1,192.168.10.0/24,11/18/2006 11:12:33.577112 AM,11/18/2006 11:12:33.577112 AM,3232238080,3232238335
2,2,1,192.168.12.0/24,11/18/2006 11:12:33.579998 AM,11/18/2006 11:12:33.579998 AM,3232238592,3232238847
2,3,1,192.168.13.0/24,11/18/2006 11:12:33.582670 AM,11/18/2006 11:12:33.582670 AM,3232238848,3232239103
2,4,1,192.168.14.0/24,11/18/2006 11:12:33.585281 AM,11/18/2006 11:12:33.585281 AM,3232239104,3232239359
2,5,1,192.168.15.0/24,11/18/2006 11:12:33.587902 AM,11/18/2006 11:12:33.587902 AM,3232239360,3232239615
2,6,1,192.168.16.0/24,11/18/2006 11:12:33.590869 AM,11/18/2006 11:12:33.590869 AM,3232239616,3232239871
2,7,1,192.168.17.0/24,11/18/2006 11:12:33.593850 AM,11/18/2006 11:12:33.593850 AM,3232239872,3232240127
2,8,1,192.168.18.0/24,11/18/2006 11:12:33.596712 AM,11/18/2006 11:12:33.596712 AM,3232240128,3232240383
2,9,1,192.168.19.0/24,11/18/2006 11:12:33.599391 AM,11/18/2006 11:12:33.599391 AM,3232240384,3232240639
2,10,1,192.168.20.0/24,11/18/2006 11:12:33.602037 AM,11/18/2006 11:12:33.602037 AM,3232240640,3232240895
2,11,1,192.168.21.0/24,11/18/2006 11:12:33.604900 AM,11/18/2006 11:12:33.604900 AM,3232240896,3232241151
2,12,1,131.136.21.0/24,11/18/2006 11:12:33.607507 AM,11/18/2006 11:12:33.607507 AM,2206733568,2206733823
2,13,1,131.132.11.0/24,11/18/2006 11:12:33.610247 AM,11/18/2006 11:12:33.610247 AM,2206468864,2206469119
2,14,1,131.136.51.0/24,11/18/2006 11:12:33.612911 AM,11/18/2006 11:12:33.612911 AM,2206741248,2206741503
2,15,1,131.132.200.0/24,11/18/2006 11:12:33.615634 AM,11/18/2006 11:12:33.615634 AM,2206517248,2206517503
3,1,1,131.136.22.0/24,11/18/2006 11:12:33.618373 AM,11/18/2006 11:12:33.618373 AM,2206733824,2206734079
3,2,1,10.136.20.0/24,11/18/2006 11:12:33.621004 AM,11/18/2006 11:12:33.621004 AM,176690176,176690431
3,3,1,10.136.21.0/24,11/18/2006 11:12:33.623620 AM,11/18/2006 11:12:33.623620 AM,176690432,176690687
4,1,1,131.136.20.0/24,11/18/2006 11:12:33.626252 AM,11/18/2006 11:12:33.626252 AM,2206733312,2206733567
5,1,1,131.132.12.0/24,11/18/2006 11:12:33.629253 AM,11/18/2006 11:12:33.629253 AM,2206469120,2206469375
5,2,1,10.132.10.0/24,11/18/2006 11:12:33.631861 AM,11/18/2006 11:12:33.631861 AM,176425472,176425727
5,3,1,10.132.11.0/24,11/18/2006 11:12:33.638927 AM,11/18/2006 11:12:33.638927 AM,176425728,176425983
6,1,1,131.132.10.0/24,11/18/2006 11:12:33.641779 AM,11/18/2006 11:12:33.641779 AM,2206468608,2206468863
7,1,1,131.136.52.0/24,11/18/2006 11:12:33.644488 AM,11/18/2006 11:12:33.644488 AM,2206741504,2206741759
7,2,1,10.136.50.0/24,11/18/2006 11:12:33.647106 AM,11/18/2006 11:12:33.647106 AM,176697856,176698111
7,3,1,10.136.51.0/24,11/18/2006 11:12:33.649737 AM,11/18/2006 11:12:33.649737 AM,176698112,176698367

DN0678
Issue 4/2: 06 February 2012

```
8,1,1,131.136.50.0/24,11/18/2006 11:12:33.652475 AM,11/18/2006 11:12:33.652475
AM,2206740992,2206741247
# zonerule
# SELECT z.*, a.ip, a.name FROM ZONERULE z, ASSET a WHERE z.serviceid=a.serviceid;
table=zonerule
headers=SERVICEID,ZONEID,RULEID,RULEORDER,CREATED,MODIFIED,PERMIT,FROMSU
BNETID,FROMIP,STARTFROMIP,ENDFROMIP,FROMPORT,TOSUBNETID,TOIP,STARTTOIP,
ENDTOIP,TOPORT,PROTOCOL,RULETEXT,IP,NAME
68,7,1,1,11/20/2006 7:36:03.312767 PM,11/20/2006 7:36:03.312767
PM,Y,1,*,0,4294967295,,1,131.136.50.0/24,2206740992,2206741247,,tcp,Test
Rule,131.136.52.1,fw-Halifax-131.136.52.0/24
68,7,2,2,11/20/2006 7:36:03.316203 PM,11/20/2006 7:36:03.316203
PM,Y,1,131.136.50.0/24,2206740992,2206741247,,1,*,0,4294967295,,tcp,Test
Rule,131.136.52.1,fw-Halifax-131.136.52.0/24
68,7,3,3,11/20/2006 7:36:03.319192 PM,11/20/2006 7:36:03.319192
PM,N,1,*,0,4294967295,,1,*,0,4294967295,,tcp,Test Rule,131.136.52.1,fw-Halifax-
131.136.52.0/24
51,3,1,1,11/20/2006 7:36:03.404806 PM,11/20/2006 7:36:03.404806
PM,Y,1,*,0,4294967295,,1,131.136.20.25,2206733337,2206733337,80,tcp,zone scenario
1,131.136.22.1,fw-Ottawa-131.136.22.0/24
51,3,2,2,11/20/2006 7:36:03.407927 PM,11/20/2006 7:36:03.407927
PM,N,1,*,0,4294967295,,1,*,0,4294967295,,*,zone scenario 1,131.136.22.1,fw-Ottawa-
131.136.22.0/24
60,5,1,1,11/20/2006 7:36:03.410897 PM,11/20/2006 7:36:03.410897
PM,Y,1,*,0,4294967295,,1,131.132.10.20,2206468628,2206468628,80,tcp,zone scenario
2,131.132.12.1,fw-Corp-131.132.12.0/24
60,5,2,2,11/20/2006 7:36:03.413916 PM,11/20/2006 7:36:03.413916
PM,N,1,*,0,4294967295,,1,*,0,4294967295,,*,zone scenario 2,131.132.12.1,fw-Corp-
131.132.12.0/24
16,2,1,1,11/20/2006 7:36:03.417755 PM,11/20/2006 7:36:03.417755
PM,Y,1,*,0,4294967295,,1,192.168.10.15,3232238095,3232238095,21,tcp,zone scenario
2,192.168.10.1,fw-Internet Firewall-192.168.10.0/24
16,2,2,2,11/20/2006 7:36:03.420805 PM,11/20/2006 7:36:03.420805
PM,N,1,*,0,4294967295,,1,*,0,4294967295,,*,zone scenario 2,192.168.10.1,fw-Internet Firewall-
192.168.10.0/24
```

Appendix D Sample Operations Data

```
<?xml version="1.0"?>

<jndms
  xmlns="http://jndms"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://jndms jndms.xsd">

  <operations>
    <operation>
      <name>CFIOG</name>
      <source> J6 </source>
      <category>operation</category>
      <priority>medium</priority>
      <type>domestic</type>
      <start_date>2006-01-02T00:00:00</start_date>
      <end_date>2010-12-31T23:59:00</end_date>
      <authority>J6</authority>
      <co>
        <full_name> Col John Doe </full_name>
      </co>
      <op_group>CFIOG</op_group>
      <poc>
        <first_name> J. </first_name>
        <last_name>Boggins</last_name>
      </poc>
      <risk_profile>
        <c>1.0</c>
        <i>1.0</i>
        <a>1.0</a>
      </risk_profile>
      <notes> Canadian Forces Information Operations Group </notes>
      <event>
        <id>1</id>
        <description>CFNOC assigned to monitor CND</description>
        <start_time>2007-06-12T00:00:00</start_time>
        <end_time>2010-12-31T23:59:00</end_time>
      </event>

    <unit>
```

DN0678
Issue 4/2: 06 February 2012

```
<name>CFNOC</name>
<poc>
  <first_name>J.</first_name>
  <last_name>Boggins</last_name>
</poc>
<notes></notes>
</unit>
<unit>
  <name>Contracted Services</name>
  <poc>
    <first_name>Un</first_name>
    <last_name>Known</last_name>
  </poc>
  <notes>Canadian Forces Network Operation Centre</notes>
</unit>

<plan>
  <record>
    <event>1</event>
    <unit>CFNOC</unit>
    <location>CFS Lietrim, ON, CA</location>

<provision><subnet>192.168.200.0/24</subnet><lossOfLife>>false</lossOfLife><impact>
  <c>low</c>
  <i>low</i>
  <a>low</a>
</impact>
<importance>useful</importance>
</provision>
<provision><subnet>192.168.201.0/24</subnet>
  <lossOfLife>>false</lossOfLife>
  <impact>
    <c>low</c>
    <i>low</i>
    <a>low</a>
  </impact>
  <importance>useful</importance>
</provision>
<provision><subnet>10.32.1.0/24</subnet>
  <lossOfLife>>false</lossOfLife>
  <impact>
    <c>low</c>
    <i>low</i>
```

DN0678
Issue 4/2: 06 February 2012

```

                                <a>low</a>
                                </impact>
                                <importance>useful</importance>
                                </provision>
                                </record>

                                <record>
                                <event>1</event>
                                <unit>Contracted Services</unit>
                                <location>Weir, QC, CA</location>
                                <provision><net>
                                <ip_address>192.168.130.1</ip_address>
                                </net>

                                <lossOfLife>>false</lossOfLife>

                                <impact>
                                <c>low</c>
                                <i>low</i>
                                <a>low</a>
                                </impact>
                                <importance>useful</importance>
                                </provision>

                                </record>

                                </plan>
                                <primary_location>CFS Lietrim, ON, CA</primary_location>
                                </operation>

                                </operations>

                                </jndms>
```

Appendix E Sample Vulnerability Report

```
<?xml version="1.0" encoding="UTF-8"?>

<vulnerability_scan>
  <scan_host>
    <host>
      <name> unknown </name>
      <ip> 131.136.20.20 </ip>
    </host>
  </scan_host>
  <scanner>
    <name>Nessus</name>
    <vendor>Tenable</vendor>
    <version>3.0.2</version>
  </scanner>
  <scan_type> remote </scan_type>
  <scan_vulnerabilities>
    <id/>

    <id>TST-0002-000</id>
    <id>CVE-2001-0500</id>

  </scan_vulnerabilities>
  <scan_results>
    <host_scan>
      <target_host>
        <name></name>
        <ip> 131.136.50.20 </ip>
      </target_host>
      <scan_start>Wed May 10 15:52:42 2006</scan_start>
      <scan_end>Wed May 10 15:57:24 2006</scan_end>
      <vulnerabilities>
        <vulnerability_instance>
          <vulnerability_ids>
            <id>TST-0002-000</id>
          </vulnerability_ids>
          <summary> Test hit for vulnerability </summary>
          <severity> vulnerability </severity>
          <risk> high </risk>
          <scan_category>infos</scan_category>
          <scan_family>Windows</scan_family>
          <information>
            Test script positive
          </information>
          <context_reference>
            <type> nessus plugin id </type>
            <version> $Revision: 1.25 $ </version>
            <reference>11214</reference>
          </context_reference>
          <scan_info>
            <name> </name>
            <type> remote </type>
            <port protocol="tcp" portid="80">

```



```
        <service_name>http</service_name>
      </port>
    </scan_info>
  </vulnerability_instance>
</vulnerabilities>
</host_scan>
</scan_results>
</vulnerability_scan>
```

Appendix F JSS Interface Definition (WSDL)

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<wsdl:definitions xmlns:mime="http://schemas.xmlsoap.org/wsdl/mime/"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/" xmlns:tns="http://mdacorporation.com/jndms/JSS/"
xmlns:wSDL="http://schemas.xmlsoap.org/wsdl/" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
name="JSS" targetNamespace="http://mdacorporation.com/jndms/JSS/">
<wsdl:types>
<xsd:schema targetNamespace="http://mdacorporation.com/jndms/JSS/">
  <xsd:element name="SIMEventResponse" type="xsd:string"/>
  <xsd:element name="CAPEventRequest" type="tns:Header"/>
  <xsd:element name="CAPEventResponse" type="xsd:string"/>
  <xsd:complexType name="SIMEventData">
    <xsd:sequence>
      <xsd:element name="type" type="xsd:string"/>
      <xsd:element name="sub_type" type="xsd:string"/>
      <xsd:element name="source_ip" type="xsd:string"/>
      <xsd:element name="target_ip" type="xsd:string"/>
      <xsd:element name="sensor_ip" type="xsd:string"/>
      <xsd:element name="sensor_event_id" type="xsd:string"/>
      <xsd:element name="sensor_cve_id" type="xsd:string"/>
      <xsd:element name="base_priority" type="xsd:string"/>
      <xsd:element name="sim_priority" type="xsd:string"/>
      <xsd:element name="correlation_cve" type="xsd:string"/>
      <xsd:element name="sensor_time" type="xsd:string"/>
    </xsd:sequence>
  </xsd:complexType>
  <xsd:complexType name="EIMEventData">
    <xsd:annotation>
      <xsd:documentation>
        EIM Message Notes:
      </xsd:documentation>
    </xsd:annotation>
    <xsd:sequence>
      <xsd:element name="source_id" type="xsd:string"/>
      <xsd:element name="type" type="xsd:string">
        <xsd:annotation>
          <xsd:documentation>
            </xsd:documentation>
        </xsd:annotation>
      </xsd:element>
    </xsd:sequence>
  </xsd:complexType>
</xsd:schema>
</wsdl:types>
</wsdl:definitions>
```

DN0678
Issue 4/2: 06 February 2012

```
</xsd:element>
<xsd:element name="ip_address" type="xsd:string"/>
<xsd:element name="name" type="xsd:string"/>
<xsd:element name="label" type="xsd:string"/>
<xsd:element name="uuid" type="xsd:string"/>
<xsd:element name="class" type="xsd:string"/>
<xsd:element name="create_date" type="xsd:string"/>
<xsd:element name="source_inst" type="xsd:string"/>
<xsd:element name="zone_id" type="xsd:string"/>
<xsd:element name="event_id" type="xsd:string"/>
<xsd:element name="severity" type="xsd:string"/>
<xsd:element name="status" type="xsd:string"/>
<xsd:element name="alt_ip" type="xsd:string"/>
<xsd:element name="alt_name" type="xsd:string"/>
<xsd:element name="alt_label" type="xsd:string"/>
<xsd:element name="alt_uuid" type="xsd:string"/>
<xsd:element name="alt_class" type="xsd:string"/>
</xsd:sequence>
</xsd:complexType>
<xsd:complexType name="VulnerabilityDefinitionData">
  <xsd:sequence>
    <xsd:element name="source_CVE" type="xsd:string"/>
    <xsd:element name="xmlNVD_CVE" type="xsd:string">
      <xsd:annotation>
        <xsd:documentation>This is the XML from CVE
See: http://nvd.nist.gov/ and http://cve.mitre.org/
      </xsd:documentation>
    </xsd:annotation>
  </xsd:element>
</xsd:sequence>
</xsd:complexType>
<xsd:complexType name="MalwareDefinitionRequest">
  <xsd:sequence>
    <xsd:element name="header" type="tns:Header"/>
    <xsd:element name="source_CME" type="xsd:string"/>
    <xsd:element name="xmlNVD_CME" type="xsd:string">
      <xsd:annotation>
        <xsd:documentation>
          This is the XML for CME (Common Malware
          Enumeration) See: http://cme.mitre.org/
        </xsd:documentation>
      </xsd:annotation>
    </xsd:element>
  </xsd:sequence>
</xsd:complexType>
```

DN0678
Issue 4/2: 06 February 2012

```
</xsd:annotation>
</xsd:element>

</xsd:sequence>
</xsd:complexType>

<xsd:complexType name="VulnerabilityScanData">
  <xsd:sequence>
    <xsd:element name="Source" type="xsd:string"/>
    <xsd:element name="VulnerabilityScanReport" type="xsd:string">
      <xsd:annotation>
        <xsd:documentation>This is the XML data for the scan report.
      </xsd:documentation>
    </xsd:documentation>
  </xsd:annotation>
</xsd:element>
  <xsd:element name="VulnerabilityReference" type="xsd:string"/>
</xsd:sequence>
</xsd:complexType>

<xsd:complexType name="TopologyRequest">
  <xsd:sequence>
    <xsd:element name="header" type="tns:Header"/>
    <xsd:element name="TopologySource" type="xsd:string">
      </xsd:element>
    <xsd:element name="TopologyXML" type="xsd:string"/>
  </xsd:sequence>
</xsd:complexType>

<xsd:complexType name="CAPEventRequest">
  <xsd:sequence>
    <xsd:element name="header" type="tns:Header"/>
    <xsd:element name="event_data" type="xsd:string">
      <xsd:annotation>
        <xsd:documentation>The event data is the XML for a CAP event.
      </xsd:documentation>
    </xsd:documentation>
  </xsd:annotation>
  </xsd:documentation>
</xsd:annotation>
</xsd:element>
</xsd:sequence>
</xsd:complexType>

<xsd:complexType name="EIMXMLEventRequest">
  <xsd:sequence>
```

DN0678
Issue 4/2: 06 February 2012

```
<xsd:element name="header" type="tns:Header"/>
  <xsd:element name="event_data" type="xsd:string"/>
</xsd:sequence>
</xsd:complexType>
<xsd:complexType name="OperationReportRequest">
  <xsd:sequence>
    <xsd:element name="header" type="tns:Header"/>
    <xsd:element name="operation_data" type="xsd:string">
      <xsd:annotation>
        <xsd:documentation>This is a JNDMS xml file.
</xsd:documentation>
      </xsd:annotation>
    </xsd:element>
  </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="SafeguardDescription">
  <xsd:sequence>
    <xsd:element name="header" type="tns:Header"/>
    <xsd:element name="safeguard_data" type="xsd:string">
      <xsd:annotation>
        <xsd:documentation>This is a JNDMS xml file.
</xsd:documentation>
      </xsd:annotation>
    </xsd:element>
  </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="AssetReport">
  <xsd:sequence>
    <xsd:element name="header" type="tns:Header"/>
    <xsd:element name="asset_data" type="xsd:string">
      <xsd:annotation>
        <xsd:documentation>The format of the asset data string depends on
the source of the data. Currently the
only source supported is the database write from Unicenter (source=ca)
</xsd:documentation>
      </xsd:annotation>
    </xsd:element>
  </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="DefensivePostureRequest">
  <xsd:sequence>
    <xsd:element name="header" type="tns:Header"/>
    <xsd:element name="data" type="tns:DefensivePostureData"/>
  </xsd:sequence>
</xsd:complexType>
```

DN0678
Issue 4/2: 06 February 2012

```
</xsd:sequence>
</xsd:complexType>
<xsd:complexType name="DSSCallRequest">
  <xsd:sequence>
    <xsd:element name="header" type="tns:Header"/>
    <xsd:element name="call" type="xsd:string"/>
    <xsd:element name="src_ip" type="xsd:string"/>
    <xsd:element name="target_ip" type="xsd:string"/>
    <xsd:element name="target_port" type="xsd:string"/>
    <xsd:element name="service_id" type="xsd:string"/>
    <xsd:element name="attack_protocol" type="xsd:string"/>
    <xsd:element name="incident_id" type="xsd:string"/>
    <xsd:element name="vulnerability_id" type="xsd:string"/>
    <xsd:element name="vulnerability_id_type" type="xsd:string"/>
  </xsd:sequence>
  </xsd:complexType>
  <xsd:element name="out" type="xsd:string"/>
  <xsd:element name="out1" type="xsd:string"/>
  <xsd:element name="out2" type="xsd:string"/>
  <xsd:element name="out3" type="xsd:string"/>
  <xsd:element name="out4" type="xsd:string"/>
  <xsd:element name="out5" type="xsd:string"/>

  <xsd:complexType name="Header">
    <xsd:sequence>
      <xsd:element name="description" type="xsd:string"/>
      <xsd:element name="stageWait" type="xsd:string"/>
      <xsd:element name="sourceID" type="xsd:string"/>
      <xsd:element name="sourceAddress" type="xsd:string">
        </xsd:element>
      <xsd:element name="gatewayID" type="xsd:string"/>
      <xsd:element name="eventID" type="xsd:string"/>
      <xsd:element name="targetAddress" type="xsd:string">
        </xsd:element>
    </xsd:sequence>
  </xsd:complexType>

  <xsd:complexType name="SIMEventRequest">
    <xsd:sequence>
      <xsd:element name="info" type="tns:Header"/>
    </xsd:sequence>
  </xsd:complexType>

```

DN0678
Issue 4/2: 06 February 2012

```
        <xsd:element name="data" type="tns:SIMEEventData"/>
    </xsd:sequence>
</xsd:complexType>

<xsd:complexType name="DefensivePostureData">
    <xsd:sequence>
        <xsd:element name="action" type="xsd:string"/>
        <xsd:element name="parameters" type="xsd:string"/>
        <xsd:element name="content" type="xsd:string"/>
    </xsd:sequence>
</xsd:complexType>

<xsd:complexType name="EIMEventRequest">
    <xsd:sequence>
        <xsd:element name="header" type="tns:Header"/>
        <xsd:element name="data" type="tns:EIMEEventData"/>
    </xsd:sequence>
</xsd:complexType>

<xsd:complexType name="VulnerabilityDefinitionRequest">
    <xsd:sequence>
        <xsd:element name="header" type="tns:Header"/>
        <xsd:element name="data" type="tns:VulnerabilityDefinitionData"/>
    </xsd:sequence>
</xsd:complexType>

<xsd:complexType name="VulnerabilityScanRequest">
    <xsd:sequence>
        <xsd:element name="header" type="tns:Header"/>
        <xsd:element name="data" type="tns:VulnerabilityScanData"/>
    </xsd:sequence>
</xsd:complexType>
</xsd:schema>
</wsdl:types>
<wsdl:message name="SIMEventResponse">
    <wsdl:part element="tns:SIMEventResponse" name="SIMEventResponse">
    </wsdl:part>
</wsdl:message>
<wsdl:message name="CAPEventResponse">
    <wsdl:part element="tns:CAPEventResponse" name="CAPEventResponse">
    </wsdl:part>
</wsdl:message>
```

DN0678
Issue 4/2: 06 February 2012

```
<wsdl:message name="CAPEventRequest">
  <wsdl:part name="CAPEventRequest" type="tns:CAPEventRequest">
  </wsdl:part>
</wsdl:message>

<wsdl:message name="SIMEventRequest">
  <wsdl:part name="SIMEventRequest" type="tns:SIMEventRequest">
  </wsdl:part>
</wsdl:message>

<wsdl:message name="EIMEventResponse">
  <wsdl:part name="EIMEventResponse" type="xsd:string"/>
</wsdl:message>

<wsdl:message name="EIMEventRequest">
  <wsdl:part name="EIMEventRequest" type="tns:EIMEventRequest"/>
</wsdl:message>

<wsdl:message name="VulnerabilityDefinitionResponse">
  <wsdl:part name="VulnerabilityDefinitionResponse" type="xsd:string">
  </wsdl:part>
</wsdl:message>

<wsdl:message name="VulnerabilityDefinitionRequest">
  <wsdl:part name="VulnerabilityDefinitionRequest" type="tns:VulnerabilityDefinitionRequest">
  </wsdl:part>
</wsdl:message>

<wsdl:message name="VulnerabilityScanResponse">
  <wsdl:part name="VulnerabilityScanResponse" type="xsd:string"/>
</wsdl:message>

<wsdl:message name="VulnerabilityScanRequest">
  <wsdl:part name="VulnerabilityScanRequest" type="tns:VulnerabilityScanRequest"/>
</wsdl:message>

<wsdl:message name="MalwareDefinitionResponse">
  <wsdl:part name="MalwareDefinitionResponse" type="xsd:string">
  </wsdl:part>
</wsdl:message>

<wsdl:message name="MalwareDefinitionRequest">
  <wsdl:part name="MalwareDefinitionRequest" type="tns:MalwareDefinitionRequest">
  </wsdl:part>
</wsdl:message>

<wsdl:message name="EIMXMLEventResponse">
```


DN0678
Issue 4/2: 06 February 2012

```
<wsdl:part name="EIMXMLEventResponse" type="xsd:string"/>
</wsdl:message>
<wsdl:message name="EIMXMLEventRequest">
  <wsdl:part name="EIMXMLEventRequest" type="tns:EIMXMLEventRequest"/>
</wsdl:message>
<wsdl:message name="OperationReportResponse">
  <wsdl:part name="OperationReportResponse" type="xsd:string"/>
</wsdl:message>
<wsdl:message name="OperationReportRequest">
  <wsdl:part name="OperationReport" type="tns:OperationReportRequest"/>
</wsdl:message>
<wsdl:message name="SafeguardDefinitionResponse">
  <wsdl:part name="SafeguardDefinitionResponse" type="xsd:string">
    </wsdl:part>
</wsdl:message>
<wsdl:message name="SafeguardDefinitionRequest">
  <wsdl:part name="SafeguardDefinitionRequest" type="tns:SafeguardDescription">
    </wsdl:part>
</wsdl:message>
<wsdl:message name="AssetReportResponse">
  <wsdl:part name="AssetReportResponse" type="xsd:string"/>
</wsdl:message>
<wsdl:message name="AssetReportRequest">
  <wsdl:part name="AssetReportRequest" type="tns:AssetReport"/>
</wsdl:message>
<wsdl:message name="DefensivePostureResponse">
  <wsdl:part name="DefensivePostureResponse" type="xsd:string"/>
</wsdl:message>
<wsdl:message name="DefensivePostureRequest">
  <wsdl:part name="DefensivePostureRequest" type="tns:DefensivePostureRequest"/>
</wsdl:message>
<wsdl:message name="DSSCallResponse">
  <wsdl:part name="DSSCallResponse" type="xsd:string"/>
</wsdl:message>
<wsdl:message name="DSSCallRequest">
  <wsdl:part name="DSSCallRequest" type="tns:DSSCallRequest"/>
</wsdl:message>
<wsdl:message name="ViewRequest">
  <wsdl:part name="ViewRequest" type="xsd:string"/>
</wsdl:message>
<wsdl:message name="ViewResponse">
  <wsdl:part element="tns:out" name="ViewResponse"/>
</wsdl:message>
```

DN0678
Issue 4/2: 06 February 2012

```
</wsdl:message>
<wsdl:message name="DeleteRequest">
  <wsdl:part name="DeleteRequest" type="xsd:string"/>
</wsdl:message>
<wsdl:message name="DeleteResponse">
  <wsdl:part element="tns:out1" name="DeleteResponse"/>
</wsdl:message>
<wsdl:message name="getStatusRequest">
  <wsdl:part name="getStatusRequest" type="xsd:string"/>
</wsdl:message>
<wsdl:message name="getStatusResponse">
  <wsdl:part element="tns:out3" name="getStatusResponse"/>
</wsdl:message>
<wsdl:message name="RemoveRequest">
  <wsdl:part name="RemoveRequest" type="xsd:string"/>
</wsdl:message>
<wsdl:message name="RemoveResponse">
  <wsdl:part element="tns:out4" name="RemoveResponse"/>
</wsdl:message>
<wsdl:message name="SubmitRequest">
  <wsdl:part name="SubmitRequest" type="xsd:string"/>
</wsdl:message>
<wsdl:message name="SubmitResponse">
  <wsdl:part element="tns:out5" name="SubmitResponse"/>
</wsdl:message>
<wsdl:portType name="JSSPort">
  <wsdl:operation name="SIMEvent">
    <wsdl:input message="tns:SIMEventRequest"/>
    <wsdl:output message="tns:SIMEventResponse"/>
  </wsdl:operation>
  <wsdl:operation name="EIMEvent">
    <wsdl:documentation>
</wsdl:documentation>
    <wsdl:input message="tns:EIMEventRequest"/>
    <wsdl:output message="tns:EIMEventResponse"/>
  </wsdl:operation>
  <wsdl:operation name="CAPEvent">
    <wsdl:input message="tns:CAPEventRequest"/>
    <wsdl:output message="tns:CAPEventResponse"/>
  </wsdl:operation>

  <wsdl:operation name="VulnerabilityDefinition">
```

DN0678
Issue 4/2: 06 February 2012

```
<wsdl:input message="tns:VulnerabilityDefinitionRequest"/>
<wsdl:output message="tns:VulnerabilityDefinitionResponse"/>
</wsdl:output>
</wsdl:operation>

<wsdl:operation name="VulnerabilityScan">
  <wsdl:input message="tns:VulnerabilityScanRequest"/>
  <wsdl:output message="tns:VulnerabilityScanResponse"/>
</wsdl:operation>

<wsdl:operation name="MalwareDefinition">
  <wsdl:input message="tns:MalwareDefinitionRequest">
  </wsdl:input>
  <wsdl:output message="tns:MalwareDefinitionResponse">
  </wsdl:output>
</wsdl:operation>

<wsdl:operation name="EIMXMLEvent">
  <wsdl:input message="tns:EIMXMLEventRequest"/>
  <wsdl:output message="tns:EIMXMLEventResponse"/>
</wsdl:operation>

<wsdl:operation name="OperationReport">
  <wsdl:input message="tns:OperationReportRequest"/>
  <wsdl:output message="tns:OperationReportResponse"/>
</wsdl:operation>

<wsdl:operation name="SafeguardDefinition">
  <wsdl:input message="tns:SafeguardDefinitionRequest"/>
  <wsdl:output message="tns:SafeguardDefinitionResponse"/>
</wsdl:operation>

<wsdl:operation name="AssetReport">
  <wsdl:input message="tns:AssetReportRequest"/>
  <wsdl:output message="tns:AssetReportResponse"/>
</wsdl:operation>

<wsdl:operation name="DefensivePosture">
  <wsdl:input message="tns:DefensivePostureRequest"/>
  <wsdl:output message="tns:DefensivePostureResponse"/>
</wsdl:operation>

<wsdl:operation name="DSSCall">
  <wsdl:input message="tns:DSSCallRequest"/>
  <wsdl:output message="tns:DSSCallResponse"/>
</wsdl:operation>
</wsdl:portType>
<wsdl:portType name="JSSQueue">
```

DN0678
Issue 4/2: 06 February 2012

```
<wsdl:operation name="View">
  <wsdl:input message="tns:ViewRequest"/>
  <wsdl:output message="tns:ViewResponse"/>
</wsdl:operation>
<wsdl:operation name="getStatus">
  <wsdl:input message="tns:getStatusRequest"/>
  <wsdl:output message="tns:getStatusResponse"/>
</wsdl:operation>
<wsdl:operation name="Remove">
  <wsdl:input message="tns:RemoveRequest"/>
  <wsdl:output message="tns:RemoveResponse"/>
</wsdl:operation>
<wsdl:operation name="Submit">
  <wsdl:input message="tns:SubmitRequest"/>
  <wsdl:output message="tns:SubmitResponse"/>
</wsdl:operation>
</wsdl:portType>
<wsdl:binding name="JSSSOAP" type="tns:JSSPort">
  <soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http"/>
  <wsdl:operation name="SIMEvent">
    <soap:operation soapAction="http://mdacorporation.com/jndms/JSS/SIMEvent"/>
    <wsdl:input>
      <soap:body use="literal"/>
    </wsdl:input>
    <wsdl:output>
      <soap:body use="literal"/>
    </wsdl:output>
  </wsdl:operation>
  <wsdl:operation name="EIMEvent">
    <soap:operation soapAction="http://mdacorporation.com/jndms/JSS/EIMEvent"/>
    <wsdl:input>
      <soap:body use="literal"/>
    </wsdl:input>
    <wsdl:output>
      <soap:body use="literal"/>
    </wsdl:output>
  </wsdl:operation>
  <wsdl:operation name="CAPEvent">
    <soap:operation soapAction="http://mdacorporation.com/jndms/JSS/CAPEvent"/>
    <wsdl:input>
      <soap:body use="literal"/>
    </wsdl:input>
  </wsdl:operation>
</wsdl:binding>
</wsdl:service>
```

DN0678
Issue 4/2: 06 February 2012

```
</wsdl:input>
<wsdl:output>
    <soap:body use="literal"/>
</wsdl:output>
</wsdl:operation>

<wsdl:operation name="VulnerabilityDefinition">
    <soap:operation
soapAction="http://mdacorporation.com/jndms/JSS/VulnerabilityDefinition"/>
    <wsdl:input>
        <soap:body use="literal"/>
    </wsdl:input>
    <wsdl:output>
        <soap:body use="literal"/>
    </wsdl:output>
</wsdl:operation>

<wsdl:operation name="VulnerabilityScan">
    <soap:operation soapAction="http://mdacorporation.com/jndms/JSS/VulnerabilityScan"/>
    <wsdl:input>
        <soap:body use="literal"/>
    </wsdl:input>
    <wsdl:output>
        <soap:body use="literal"/>
    </wsdl:output>
</wsdl:operation>

<wsdl:operation name="MalwareDefinition">
    <soap:operation soapAction="http://mdacorporation.com/jndms/JSS/MalwareDefinition"/>
    <wsdl:input>
        <soap:body use="literal"/>
    </wsdl:input>
    <wsdl:output>
        <soap:body use="literal"/>
    </wsdl:output>
</wsdl:operation>

<wsdl:operation name="EIMXMLEvent">
    <soap:operation soapAction="http://mdacorporation.com/jndms/JSS/EIMXMLEvent"/>
    <wsdl:input>
        <soap:body use="literal"/>
    </wsdl:input>
    <wsdl:output>
```

DN0678
Issue 4/2: 06 February 2012

```
        <soap:body use="literal"/>
    </wsdl:output>
</wsdl:operation>
<wsdl:operation name="OperationReport">
    <soap:operation soapAction="http://mdacorporation.com/jndms/JSS/OperationReport"/>
    <wsdl:input>
        <soap:body use="literal"/>
    </wsdl:input>
    <wsdl:output>
        <soap:body use="literal"/>
    </wsdl:output>
</wsdl:operation>
<wsdl:operation name="SafeguardDefinition">
    <soap:operation soapAction="http://mdacorporation.com/jndms/JSS/SafeguardDefinition"/>
    <wsdl:input>
        <soap:body use="literal"/>
    </wsdl:input>
    <wsdl:output>
        <soap:body use="literal"/>
    </wsdl:output>
</wsdl:operation>
<wsdl:operation name="AssetReport">
    <soap:operation soapAction="http://mdacorporation.com/jndms/JSS/AssetReport"/>
    <wsdl:input>
        <soap:body use="literal"/>
    </wsdl:input>
    <wsdl:output>
        <soap:body use="literal"/>
    </wsdl:output>
</wsdl:operation>
<wsdl:operation name="DefensivePosture">
    <soap:operation soapAction="http://mdacorporation.com/jndms/JSS/DefensivePosture"/>
    <wsdl:input>
        <soap:body use="literal"/>
    </wsdl:input>
    <wsdl:output>
        <soap:body use="literal"/>
    </wsdl:output>
</wsdl:operation>
<wsdl:operation name="DSSCall">
    <soap:operation soapAction="http://mdacorporation.com/jndms/JSS/DSSCall"/>
    <wsdl:input>
```

DN0678
Issue 4/2: 06 February 2012

```
        <soap:body use="literal"/>
    </wsdl:input>
    <wsdl:output>
        <soap:body use="literal"/>
    </wsdl:output>
</wsdl:operation>
</wsdl:binding>
<wsdl:binding name="QueueSOAP" type="tns:JSSQueue">
    <soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http"/>
    <wsdl:operation name="View">
        <soap:operation soapAction="http://mdacorporation.com/jndms/JSS/View"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="getStatus">
        <soap:operation soapAction="http://mdacorporation.com/jndms/JSS/getStatus"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="Remove">
        <soap:operation soapAction="http://mdacorporation.com/jndms/JSS/Remove"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="Submit">
        <soap:operation soapAction="http://mdacorporation.com/jndms/JSS/Submit"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
```

DN0678
Issue 4/2: 06 February 2012

```
<soap:body use="literal"/>
</wsdl:output>
</wsdl:operation>
</wsdl:binding>
<wsdl:service name="JSS">
  <wsdl:port binding="tns:JSSSOAP" name="JNDMSPort">
    <soap:address location="http://localhost:8080/JSS/services/JNDMSPort"/>
  </wsdl:port>
  <wsdl:port binding="tns:QueueSOAP" name="JNDMSQueue">
    <soap:address location="http://localhost:8080/JSS/services/JNDMSQueue"/>
  </wsdl:port>
</wsdl:service>
</wsdl:definitions>
```


Appendix G Execution Environment

The instructions for building and deploying the application are contained in a separate document:

Appendix G - Building and Deploying JNDMS to the Execution Environment