NETRANGER™

USER'S GUIDE

Version 1.3.1

Copyright © 1997 by WheelGroup Corporation



PAN EX1049
Palo Alto Networks v. BT Americas
IPR2023-00889

WheelGroup Corporation has made every effort to ensure that this *User's Guide* is correct and accurate, but reserves the right to make changes without notice at its sole discretion at any time.

The NetRanger System described in this guide is supplied under a license and may be used only in accordance with the terms of such license, and in particular any warranty of fitness of WheelGroup Corporation products for any particular purpose is expressly excluded and in no event will WheelGroup Corporation be liable for any consequential loss.

CONFIDENTIAL. The NetRanger System software is confidential and a trade secret of WheelGroup Corporation. All use, disclosure, and/or reproduction of this software not expressly authorized in writing by WheelGroup Corporation is prohibited. WheelGroup Corporation shall prosecute all unauthorized use, disclosure, and reproduction of this software under the criminal and civil laws.

NetRanger, WheelGroup Corporation, and the WheelGroup logo are trademarks of WheelGroup Corporation. All other product names in this publication are trademarks or registered trademarks of their owners.

Copyright © 1997 by WheelGroup Corporation

All rights reserved. No part of this publication may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopy, recording, or any information storage and retrieval system, without permission in writing from the publisher.

Produced in the United States of America.

GETTING HELP

WheelGroup Technical Assistance

Refer to this *User's Guide* whenever you have a problem using the NetRanger System. If you still cannot solve the problem or if you have questions about anything not covered in this guide, you can call WheelGroup Technical Support from 8:00AM to 6:00PM, CST. This assistance is available Monday through Friday (except holidays). You can reach us in any of the following ways:

Telephone	**1-888-942-6762**
FAX	**210-494-6303**
E-Mail	**help@wheelgroup.com**

Please include the following information in your FAX or e-mail message, or have it available if you are calling (use the NetRanger Problem Report Form on the following page to help you gather this information):

- Your customer number and phone number. For e-mail messages, please include your customer number as part of the subject line for your messages.
- A description of the NSX and Director hardware and software you are using, including any network software

  To determine the version number of the Director software and daemons you are using, type the following commands:

  ```
  nrdirmap -?
  /usr/nr/nrvers
  ```

- Your operating system type and version
- A description of the problem, what you were doing when it occurred, and the exact wording of any error messages that you might have received
- If this is a security question, please include all information related to a specific alarm or attacking site.

NetRanger Problem Report Form

Before you call, please have the following information available.

Organization ID:_____ Host ID (of Problem System):_____

Point of Contact:_____ Phone Number:_____

For Director or NSX Problem

NSX Information (if applicable)

NSX Type:_____ Number of NSC devices connected to NSX:_____

NSC Type (with problem):_____

NSC Serial Number (with problem):_____

Director Information (if applicable)

Platform: HP SPARC

OV Version:_____ Director Version:_____

Problem Description

_____

_____

_____

_____

_____

For Security Incidents

Are you a monitored site? Yes No Member of WARN? Yes No

IP addresses of involved machines:_____

Do you wish to talk to WheelGroup about Intrusion Control and Response (ICR) consulting? Yes No

Alarm Type:_____

Description of security incident:_____

_____

_____

WheelGroup Corporation
1-888-942-6762
1-210-494-6303 (fax)
e-mail: help@wheelgroup.com

NETRANGER 1.3 ENHANCEMENTS

This section describes enhancements made to the 1.3 release of NetRanger with respect to the following areas:

- **Installation**
- **Director**
- **Communication**
- **NSX**

Installation

Bridge Filter Changes

In order to properly interpret copy_to and log_to packets under bridge mode all references to tl_byte have been changed to tl_data_byte in the files: first.fil, incom1.fil, and incom2.fil. All references to ip_option_present have also been commented out in incom1.fil and incom2.fil because this option is not supported in bridge mode.

Bug Fixes

- The **nrconfig** utility no longer generates an invalid default route under bridge mode.
- The **pkgadd** process no longer changes the usr:grp permissions of the /usr directory to netrangr:netrangr.
- The **pkgadd** process no longer destroys existing configuration information in /usr/nr/bin/sap.

Director

Support for NetView on IBM/AIX

The NetRanger Director now works with 4.1 and later versions of NetView running with 4.1 and later versions of the IBM/AIX operating system.

Discontinued Solaris 2.4 Support

The 1.3 version of NetRanger no longer supports SPARC Solaris version 2.4.

Improved Alarm Performance

The Director has always avoided generating duplicate alarms by scanning the OVwDB before generating an alarm icon. Prior versions of the Director incurred unnecessary processing overhead by unconditionally scanning the OVwDB. The Director now makes sure that the severity level of the alarm equals or exceeds the OpenView display threshold before it scans the OVwDB.

Improved Alarm Consolidation

Prior to the 1.3 release, the Director generated a separate alarm icon for every source port accessed by a ***DNS Request*** and for every src-dst port pair accessed by a ***TCP Port Sweep.*** The Director now generates a single alarm icon for each of these types of events. This dramatically reduces icon and OVwDB clutter as well as improving overall performance under load. Please note that while the Director generates a single icon for each of these types of events, all of the ports accessed by these services are properly logged by *loggerd.*

Single Window for Source and Destination Address Name Resolution

In previous versions of NetRanger, when the **Security→Show→Names** menu option was selected for multiple alarm icons the Director would generate a window for each alarm. The Director now generates a single window with a scrollable display of alarm names.

New Menu Functions

- The **Security→Exclude Alarms** menu option allows you to suppress all of the alarms for a given src-dst pair.
- The **Security→Show Alarm Submap(s)** menu option allows you to display all of alarms associated with a machine or collection without having to traverse its submaps.
- The **Security→Trouble Ticket** menu option allows you to generate a trouble ticket in a Remedy ARS system for the alarm(s) currently selected.
- The Director now supports the OpenView **Manage** and **Unmanage** functions.

SAP Enhancements

- FileMgmt functionality has been extended and simplified.
- Improved instrumentation tokens have been developed for *sapd* Status and Control.
- ***sapr*** queries are more easy to use and generate more meaningful results.
- *loggerd* is now able to process **heavier alarm loads** from multiple sensors by caching records in a write buffer.
- Chapter 5 of this *User's Guide* now contains new ***Installation*** and ***Tutorial*** sections for SAP.

Bug Fixes

- The **Save-To-File** feature no longer fails when an NSX Collection Name contains spaces, which are now automatically promoted to underbars.
- The *nrdlrmap* application now properly cleans up orphaned socket(s) before it tries to reestablish communication with *smid* after a failed communication has occurred.
- The Director no longer generates an error message when it tries to Unshun a network or host.

Communication

Repetitive error messages relating to misconfiguration are now consolidated in errors.postofficed. This helps minimize file system usage.

NSX

Passport in Bridge Mode

The 1.3 version of NetRanger now recognizes log_to packets transmitted by a Passport.

Bug Fixes

- NSX sensors now recognize the ***Multiplexor*** series of the BorderGuard.
- **nrstart** now invokes *managed* when started as user *netrangr* under SPARC Solaris.

Modified Signatures

Signature	Modification
2151	Increased size of allowed ICMP traffic
2000-2012	SubSigID now contains ICMP code value

New *Context* Signatures

Signature	Name
2152	ICMP Flood
3105	Sendmail Decode
4050	UDP Bomb
6200	Ident Buffer Overflow
6201	Ident Newline
6202	Ident Improper Request

New *Content* Signatures

- IFS hacks
- rlogin - froot
- ftp USER
- ftp PASS
- ftp RETR
- ftp STOR
- ftp MKD
- ftp RMD
- ftp SITE
- ftp SYST
- ftp CWD ~root
- ftp PORT
- http GET
- Bad URLs
- rexec USER
- rexec PASS
- Ident USER
- pop3 USER
- pop3 PASS
- IP Fragment
- Unknown IP Protocol
- Authentication Fail

1 OVERVIEW 1-1

What is NetRanger? 1-1
The NSX 1-2
The Communication System 1-3
The Director 1-3

Product Capabilities 1-3
NSX System 1-5
NETWORK SENSING 1-6
NETWORK PROTOCOLS 1-6
PACKET FILTER DEVICES 1-6

Performance Capabilities 1-7
NetSentry 1-7
ATTACK SIGNATURES 1-8
ATTACK RESPONSES 1-8
Alarms 1-9
Shunning 1-9
Logging 1-9
Communications Protocol 1-11
Alternate Routes 1-11
Distribution of Data 1-12
Distribution Hierarchies 1-12
Encrypted Sleeves 1-13

Director System 1-14
NSX Monitoring 1-15
NETWORK SECURITY MAPS 1-15
Icon States 1-16
NSX MANAGEMENT 1-17
NSX Data Collection 1-18
NSX Data Analysis 1-19
User-Defined Actions 1-20

2 NETRANGER PRE-INSTALLATION 2-1

Analyze the Current Network Architecture 2-1

Define All Entry and Exit Points to the Protected Network 2-1
Case 1—One Geographic Location, No Existing Internet Connection 2-2
Case 2—One Geographic Location, Existing Internet Connection 2-2
Case 3—Multiple Geographic Locations, Existing Internet Connection 2-2

Identify Current Security Measures 2-3
How Existing Firewalls Affect NetRanger Installation 2-3
How Security Filters in Existing Routers Impact NetRanger Installation 2-3

NSG BorderGuard Installation Options 2-3
Option 1—Install the BorderGuard as a Bridge 2-3
Option 2—Install the BorderGuard as the Internet Router 2-5
Options 3–6—Unable to Replace the Existing Internet Router 2-6
Option 3—A Class B Address 2-6
Option 4—An Unused Class C Address 2-7

Option 5—Unable to Replace Existing Internet Router and No Unused Class C Addresses 2-8
Option 6—Unable to Subnet The Existing Class C Address 2-9

NetRanger NSX Sensor Installation Options **2-10**
Option 1—Install the NSX Sensor on a Separate, Isolated Network 2-10
Option 2—Install the NSX Sensor on the Corporate Network 2-11
Option 3—Install the NSX Sensor on a Switched Ethernet Network 2-12

NetRanger Director Setup Options **2-13**

Physical Installation Considerations **2-13**
Power 2-13
Physical and Operational Specifications 2-13
THE BORDERGUARD 1000 2-13
Dimensions 2-13
Environment 2-13
Power Requirements 2-14
Regulations (U.S.) 2-14
EMI/RFI 2-14
THE BORDERGUARD 2000 2-14
Dimensions 2-14
Environment 2-14
Power Requirements 2-14
Regulations (U.S.) 2-14
EMI/RFI 2-15
THE NSX SENSOR 2-15
Dimensions 2-15
Environment 2-15
Power Requirements 2-15
Regulations (U.S.) 2-15
EMI/RFI 2-15

3 CONFIGURATION AND INSTALLATION **3-1**

Install and Configure the NetRanger NSX Sensor **3-1**
Position the NSX 3-1
Attach Power, Network, and Modem Cables 3-2
Access and Configure the NSX System 3-2
NETWORK ACCESS 3-2
MODEM ACCESS 3-3
SERIAL ACCESS 3-3
CONSOLE ACCESS 3-3
Logging In 3-4
Initial Configuration 3-4
PERFORM NETRANGER-SPECIFIC CONFIGURATION 3-5

Install and Configure the BorderGuard/Passport Device **3-5**
BorderGuard Installation 3-5
BorderGuard Configuration 3-5
BorderGuard Cabling and Setup 3-7
CABLE REQUIREMENTS 3-7
Passport Configuration 3-8

Install and Configure the NetRanger Director ***3-10***
NetRanger Director Software and Hardware Requirements *3-10*
SOFTWARE REQUIREMENTS 3-10
HARDWARE REQUIREMENTS 3-10
Disk Space 3-10
RAM 3-11
Pre-Installation *3-12*
HP-UX SYSTEMS AND SUN SOLARIS SYSTEMS 3-12
Installing HP-UX 10.10 or greater and Installing Solaris 2.5 or greater 3-12
Installing HP OpenView 4.1 or greater on HP-UX systems 3-12
Installing HP OpenView 4.1 or greater on Solaris 2.5 or greater 3-13
PRE-INSTALLATION FOR IBM AIX SYSTEMS 3-15
Installing AIX 4.1 or greater 3-15
Installing NetView for AIX 4.1 or greater on AIX 4.1 or greater 3-15
Director Installation *3-17*
Post-Installation *3-19*
CONFIGURING THE NETRANGER BACKGROUND PROCESSES 3-19
CONFIGURING THE NETWORK MANAGEMENT BACKGROUND PROCESSES 3-20
Disabling Daemons in HP-UX and Sun Solaris 3-20
Disabling Daemons in AIX 3-21
CONFIGURING THE USER INTERFACE FOR HP-UX AND SUN SOLARIS SYSTEMS 3-21
CONFIGURING THE USER INTERFACE FOR AIX SYSTEMS 3-24
CONFIGURING NEW USERS 3-25
Adding Users to the netrangr Group 3-25
Configuring the User Environment 3-25

Complete the NetRanger Configuration Instructions and Worksheets ***3-26***

Define the Security Policy ***3-27***
ICMP (Internet Control Message Protocol) *3-28*
TCP (Transmission Control Protocol) *3-29*
UDP (User Datagram Protocol) *3-31*

4 OPERATING NETRANGER **4-1**

Architecture ***4-2***

Basic Director Functions ***4-3***
Starting the Director *4-3*
Starting the NetRanger Background Processes *4-3*
Starting the network management background processes *4-3*
Starting the Network Management User Interface *4-4*
Stopping the Director *4-4*
Stopping the Network Management User Interface *4-4*
Stopping the NetRanger background processes *4-5*
Checking the Status of the Director Processes *4-5*

Understanding the Director's Submap Hierarchy ***4-6***

Adding Entities ***4-10***
Manually Adding an NSX Machine Symbol *4-10*
Manually Adding an Application Symbol *4-12*
Manually Adding the Director Machine *4-13*

Manually Adding an NSX Collection 4-14
Modifying Entity Attributes 4-15
NSX Collection Attributes 4-15
MACHINE ATTRIBUTES 4-15
APPLICATION ATTRIBUTES 4-16
ALARM ATTRIBUTES 4-18
Deleting Entities 4-19
How to Partition a Map 4-20
Changing Map Configuration Parameters 4-21
NetRanger Director User Interface Menu Functions 4-23
Remotely Configuring NetRanger Daemons 4-23
Viewing Alarm Context Information 4-23
Viewing Event Lists 4-24
Viewing Database Status and Configuration 4-25
Resolving an Alarm's IP Addresses 4-26
Determining the Version of a Remote NetRanger Daemon 4-26
Shunning IP Addresses and Class C IP Networks 4-26
Unshunning IP Addresses and Class C IP Networks 4-27
Saving Object Data to a File 4-27
FINDING OUT ABOUT NRDIRMAP 4-27
Creating a Trouble Ticket 4-28
Changing IP Addresses, Hostnames, and NetRanger IDs 4-28
Changing Registration Files 4-29
-A, ALARM CONSOLIDATION THRESHOLD 4-30
-C, CRITICAL VALUE THRESHOLD 4-30
-D, DISABLE NRDIRMAP BY DEFAULT FOR NEW MAPS 4-31
-F, FORCE FULL SYNCHRONIZATION 4-31
-I, SECONDS TO BE IDLE BEFORE SLEEP 4-32
-K, KEEP SYMBOL COPIES DURING DELETE 4-32
-M, MARGINAL VALUE THRESHOLD 4-32
-P, PROPAGATE TO IP MAP 4-33
-S, SECURE MAP CREATION 4-33
-T, TRACING ENABLED 4-34
Command Line Parameter Examples 4-35
Changing the Number of Events Displayed in Event List 4-35
Changing Symbol, Object, and Submap Characteristics 4-36
OBJECT MODIFICATIONS 4-36
SYMBOL MODIFICATIONS 4-36
SUBMAP MODIFICATIONS 4-37
SEARCHING FOR SYMBOLS 4-37
SETTING THE HOME SUBMAP 4-38
CHANGING A SUBMAP BACKGROUND 4-38
REPOSITIONING SYMBOLS ON A SUBMAP 4-39
HIDING SYMBOLS 4-39
CHANGING THE STATUS PROPAGATION SCHEMES 4-40
CHANGING APPEARANCES, FONTS, WINDOW SIZES, COLORS, ETC. 4-40
CREATING AND USING MULTIPLE MAPS 4-41
USING READ-ONLY MAPS 4-43
LIMITING ACCESS TO NETRANGER DIRECTOR SECURITY INFORMATION 4-44
DISABLING NRDIRMAP 4-44

SETTING USER GROUPS 4-45
SETTING MAP GROUPS AND PERMISSIONS 4-46
Set-up 4-47
Create the "secinfo" map 4-47
Set the "secinfo" map permissions 4-47
Create the "nosecinfo" map 4-47
Set the "nosecinfo" map permissions 4-47

nrConfigure 4-48
Overview 4-48
Architecture 4-48
Starting nrConfigure 4-49
Quitting nrConfigure 4-49
Help with nrConfigure 4-50
Configuring an Application with nrConfigure 4-50
NetRanger Token Order in the nrConfigure GUI 4-51

eventd 4-55
Overview 4-55
Using eventd to set up an e-mail notification system 4-56
BASIC SETUP 4-56
Set Up the Event Script's Configuration File 4-56
Set Up eventd's Configuration File 4-58
Set Up smid's Configuration File 4-58
Set Up eventd to Start 4-58
Advanced Setup 4-59
MONITORING MULTIPLE ORGANIZATIONS 4-59
CHANGING THE EVENT ERROR NOTIFICATION USER 4-60

Event Signatures 4-60
New Signatures in the 1.3.1 Release 4-61
Modified Signatures 4-62
TCP/IP Event Signatures 4-63
CONTEXT-BASED SIGNATURES 4-63
CONTENT-BASED SIGNATURES 4-64
IP Options Events 4-65
ICMP Events 4-67
Regular Expression-Based Signatures 4-78
NetSentry-Based Signatures 4-81

5 THE SECURITY ANALYSIS PACKAGE 5-1

High Level Overview 5-2
Collection 5-3
Management 5-3
Analysis 5-3

Functional Overview 5-5
Collection 5-6
EVENT ALARM DATA 5-6
Fixed Alarm Data 5-6
Optional Alarm Data 5-6
LOG FILE SERIALIZATION 5-7
FAIL-SAFE FEATURES 5-8

Runaway Serialization 5-8
Loss of Files from Overwriting 5-9
Manual Serialization 5-9
Management *5-9*
SAPD 5-9
Conditions 5-10
Actions 5-10
SAPX 5-11
Analysis *5-12*
SAPR 5-12
Installation ***5-13***
Before You Install the Security Analysis Package *5-13*
LOGGING PROFILE 5-13
SERIALIZATION THRESHOLDS 5-14
FILE MANAGEMENT THRESHOLDS 5-14
TARGET ARCHIVE AND DUMP DESTINATIONS 5-15
NOTIFICATION INTERVAL, CONDITIONS, AND DESTINATION 5-15
Installing and Configuring SAP on an NSX *5-16*
Installing and Configuring SAP on a Director *5-20*
ORACLE DBMS SETUP 5-21
Database Server Setup 5-21
Database Schema Setup 5-21
Database Load Setup 5-22
Database Reports Setup and Customization 5-22
NON-ORACLE DBMS SETUP 5-24
SAP Tutorial ***5-26***
How to Use Instrumentation *5-26*
VIEWING FILEMGMT INSTRUMENTATION 5-26
USING VARSIZE 5-29
USING FM_ACTION 5-30
Triggers *5-30*
HOW TO CONFIGURE A TRIGGER 5-33
How to Change the PollingDelay *5-33*
SAP Reference ***5-34***
Thresholds and Triggers *5-34*
DO YOU HAVE ENOUGH SPACE? 5-34
DO YOU HAVE ENOUGH ROOM LEFT OVER FOR ARCHIVING? 5-34
HOW DO YOU WANT YOUR DATA BATCHED? 5-35
sapr *5-35*
REPORTS 5-35
CUSTOMIZING QUERIES 5-36
SQL DATABASE INFORMATION 5-37
Default Schemas 5-37
Scripts and skel *5-41*
ALTERNATE LOADING PROCEDURES 5-42
Defining what gets Loaded 5-42
Changing the Target Database from Oracle to Remedy ARS 5-42
ALTERNATE DATABASE SCHEMAS AND DESTINATIONS 5-42
NSOC Operations *5-43*
DBA Operations *5-44*
FIX STORAGE PROBLEMS 5-44

PURGING DATA 5-44
File Management Tokens *5-46*

A TROUBLESHOOTING A-1
Director Not Running *A-2*
Director Running *A-4*
Connectivity *A-6*
NSX *A-7*
Oracle *A-8*
Security Analysis Package *A-10*

B FILTERS AND BRIDGE CONFIGURATION B-1
Standard Files ***B-1***
Filter Templates ***B-3***
NetRanger and Bridging ***B-7***
Filters *B-9*
DPF Files ***B-10***
Bridging and DPF *B-13*
NetRanger Support for Passport Bridging ***B-14***

C DEFAULT NETRANGER ALARM SETTINGS C-1

D DBMS REQUIREMENTS AND SETUP D-1
Oracle Install and Setup ***D-1***
Plan your Oracle Environment *D-1*
Install/Mount the Oracle Product *D-2*
LOCAL D-2
REMOTE D-2
Remote homogeneous D-2
Remote heterogeneous D-2
UNIX Setup of Oracle Environment Variables *D-3*
REMOTE OR LOCAL D-3
LOCAL ONLY D-4
TTY PROBLEMS (HP-UX) D-4
Verify Generic Connectivity *D-4*
LOCAL D-5
REMOTE D-5
Create an Oracle User *D-6*
Verify NetRanger–Oracle Connectivity *D-7*
Create Tables *D-7*
Setup for Remedy ARS (Optional) *D-7*
Non-Oracle Install and Setup ***D-7***
DBMS Setup Checklist ***D-8***
Install and Configure Oracle *D-8*

NetRanger–Oracle Setup *D-8*
nr.sapd Setup *D-8*

E THE NSX FILE STRUCTURE E-1

~/bin ***E-2***
Daemon Applications *E-2*
CONFIGD E-3
EVENTD E-3
LOGGERD E-3
MANAGED E-4
POSTOFFICED E-4
SAPD E-5
SMID E-5
SENSORD E-5
Configuration Commands *E-6*
NRGET E-6
NRGETBULK E-6
NRSET E-6
NRUNSET E-6
NREXEC E-7
System Commands *E-7*
INSTALL E-7
NRSTART E-7
NRSTOP E-7
NRSTATUS E-7

~/classes ***E-8***

~/etc ***E-8***
Daemon Configuration Files *E-8*
MANAGED.CONF E-9
SENSORD.CONF E-10
General E-10
Event Specification E-11
Strings To Look For E-11
Matched String and Event Levels E-12
TCP/UDP Ports and Event Levels E-12
Policy Violations E-12
Excluded Events E-13
SMID.CONF E-13
NSX SYSTEM FILES E-14
ORGANIZATIONS E-14
HOSTS E-15
SERVICES E-15
AUTHS E-16
DESTINATIONS E-16
ROUTES E-17
DAEMONS E-17
SIGNATURES E-18

~/skel ***E-18***

~/tmp ***E-19***

~/var *E-19*
Log Files *E-19*
EVENT FILE E-19
Session Log Files *E-21*

F NETRANGER HARDWARE COMPONENTS F-1

NSX Sensor Hardware Components ***F-1***

NSX Versions ***F-2***
NSX 1000 ..... *F-2*
NSX 2000 ..... *F-2*
NSX 5000 ..... *F-3*

NSG BorderGuard Hardware Components ***F-3***

G UNINSTALLING NETRANGER G-1

H NETRANGER managed FAQ H-1

OVERVIEW

1

NetRanger's Components and Capabilities

What is NetRanger?

NetRanger is a real-time network security management system that detects, analyzes, responds to, and deters unauthorized network activity. The NetRanger architecture supports large-scale information protection via centralized monitoring and management of remote dynamic packet filtering devices that plug into TCP/IP networks. Communication is maintained via WheelGroup Corporation's (WGC) proprietary secure communications architecture. Network activity can also be logged for more in-depth analysis.

As shown in Figure 1.1, NetRanger 1.3.1 consists of the following core systems and subsystems:

- **Network Security eXchange (NSX)**
  - Packet Filtering Device(s)
  - Sensor
- **Communications System**
  - Post Office(s)
  - Encrypted Sleeve
- **Director**
  - Security Management Interface
  - Security Analysis Package

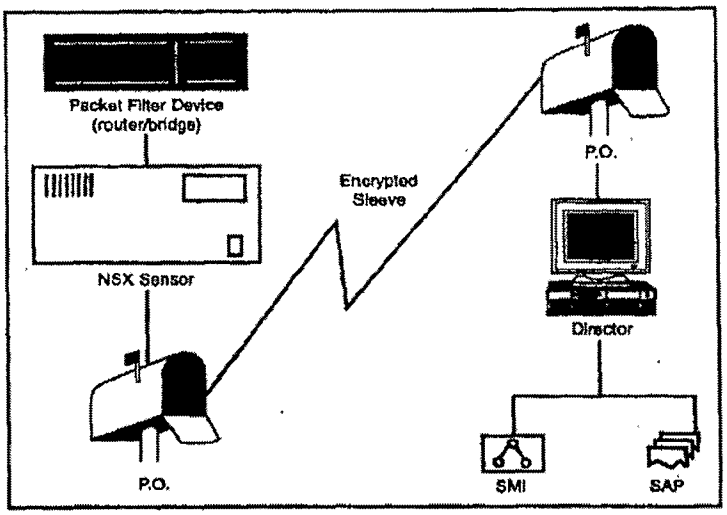


Figure 1.1: Basic NetRanger Components

The NSX

The **NSX** is the sensing and management component of the NetRanger System that resides on a corporate network. It communicates with one or more remote Director systems via the Post Office network communications system. The NSX currently operates with IP networks and supports many hardware and software configuration options.

The **Packet Filtering Device** is a router or bridge that plugs into a corporation's network at a point of entry to other networks. The security policy installed on this device determines what subset of network traffic will be routed to the NetRanger Sensor.

The **Sensor** subsystem contains NetRanger's real-time intrusion detection and content assessment logic. The intrusion detection engine recognizes and responds to attacks, such as sendmail, ping sweeps, IP source routing and spoofing, FTP, and Telnet abuse, and SATAN scans. Sensor analyses produce data streams of IP packets and event records that are either dumped into local session log files or sent on to the Post Office for remote delivery to a Director system. The Sensor also accepts intrusion response and reconfiguration information from Director systems.

The Communication System

The **post office** subsystem provides the communication backbone for remote monitoring and NSX systems management. All communication is based on a proprietary, connection-based protocol that can switch between alternate routes in order to maintain the point-to-point connections specified in its routing tables. All messages are routed based on a three-part address that includes organization, host, and application identifiers.

The **Encrypted Sleeve** secures data transmission between remote protected networks. Encrypted sleeves are currently implemented via the Data Privacy Facility (DPF) that comes with Network Systems Group's (NSG) BorderGuard packet filter devices.

The Director

The **Director** provides monitoring and analysis services to NetRanger, and communicates with one or more NSX systems via the communication system. The Director contains two basic subsystems, the Security Management Interface (SMI) and the Security Analysis Package (SAP).

The **SMI** is a collection of GUIs and tools that help monitor and respond to security events at one or more NSX locations. The SMI integrates with network management applications (such as HP OpenView®). Whereas the SMI is focused primarily on real-time security event management, data analysis is supported by the SAP.

The **SAP** is a set of data analysis tools that analyzes NSX data independently of SMI activities. The SAP consists of three basic components: data collection, data management, and data analysis. Although all components can be easily integrated into an existing SMI platform, WheelGroup recommends that all data be exported onto a separate database server. In this way, the SMI and SAP components can be configured, secured, and tuned independently.

Product Capabilities

One of the attributes that make NetRanger the preeminent intrusion detection and response system is the way in which it integrates many tried and true network security technologies. Most network security applications fall into one of two categories: **firewalls** or **network monitoring**. Both of these technologies suffer from shortcomings not found in NetRanger.

Firewalls represent the most common form of network security, and they gained popularity in part because of their relatively simple host-based installation and friendly graphical user interfaces. The biggest problem with firewalls, however, is that they rely upon *proxy services* to enforce an organization's security policies. Proxy services erect *static* barriers that frequently block legitimate as well as illegitimate activity. This puts

pressure on system administrators to open pathways, which frequently makes firewalls vulnerable to the very events they are supposed to guard against. Because most firewalls are host-based solutions, administration of more than one firewall at a time is also difficult. The system overhead associated with proxy services also prevents most firewalls from being able to scale beyond Ethernet speeds.

Although network monitoring tools can detect unauthorized activities without having to erect barriers to entry, administrators typically have to sift through audit logs after the fact in order to find security breaches. In many instances, systems have been compromised by the time the activity has been detected. Both of these approaches also tend to only look at incoming network traffic.

NetRanger enforces an organization's security policy via **real-time response and detection of intrusive events** without having to erect static barriers. NetRanger's secure communication architecture also allows command and control, as well as system information, to be distributed securely across heterogeneous networks. This section describes these capabilities in a top-down fashion within the context of the underlying architecture.

One of the fundamental design principles of NetRanger is that the *services* required by each of the subsystems be broken apart into their atomic operational components, or *daemons*, which are diagrammed in Figure 1.2. For example, the NetRanger daemon that logs events is totally separate from the ones that perform network sensing and device management. The reasons for this approach are **speed, durability, scalability**, and **independence.**

Each of NetRanger's daemon services is purpose-built for a specific task. This makes it possible to optimize each service without compromising the functionality of the other services. Purpose-built components also tend to be more durable than systems that manage multiple tasks, and are easier to debug and upgrade.

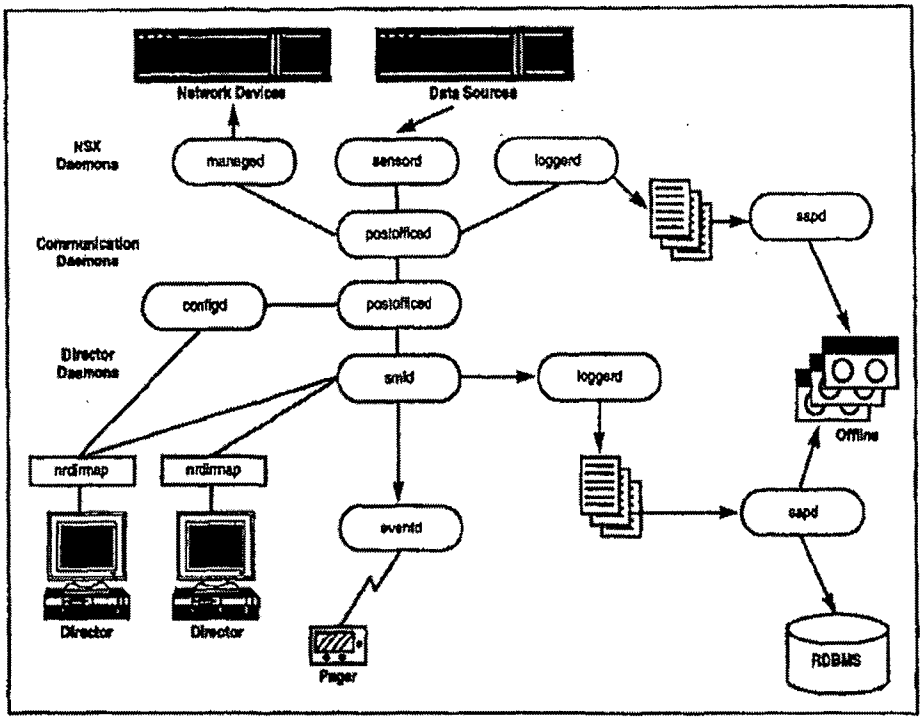


Figure 1.2: NetRanger 1.3.1 Architecture

NSX System

The NSX's capabilities are best described as **network sensing, device management,** and **session logging,** implemented respectively by the *sensord, managed,* and *loggerd* daemons diagrammed in Figure 1.2.

Although NetRanger is frequently described as an *Intrusion Detection* system, it also looks for a variety of suspicious activities that precede unauthorized events. An NSX will detect and report a ping sweep of a network. Although not truly intrusive, a ping sweep is frequently a precursor to unauthorized activity. The NSX system therefore looks for network **patterns of misuse** based on a variety of different **attack signatures.**

Network Sensing

Patterns of misuse are identified by two basic types of network signatures: **context** and **content**. Context-based signatures deal with the *state* of a transmission as defined by the structure of packet headers, and content-based signatures focus on *what* is being transported—the binary data. Context-based signatures tend to be more complex than content-based ones and the steps required to identify them are also complex and proprietary. As a consequence, context-based signatures are embedded within the NSX sensor subsystem, whereas content-based signatures are configurable and can be added dynamically at runtime.

Network Protocols

The sensor subsystem currently works with **TCP/IP**. Future releases of NetRanger will support other network protocols, such as Novell's IPX/SPX.

Packet Filter Devices

The only type of packet filter devices the sensor subsystem currently works with are the **BorderGuard** and **Passport** devices from Network Systems Group (NSG). These packet filter devices play a key role in the success of the NSX system.

In addition to serving as high-speed IP data sources, all of these devices

- can be **reconfigured on the fly**,
- support a common **NetSentry interface**,
- can be deployed as **bridges** as well as **routers**, and
- can maintain **Virtual Private Network** (VPN) connections.

Because these devices can be reconfigured on the fly, NetRanger can dynamically **shun as well as detect** suspicious and unauthorized network activity. The common command and control interface provided by NetSentry allows one NSX to support all three devices. Please note that these devices can operate as bridges as well as routers, which means that an NSX can be deployed in a network behind existing devices, such as Cisco routers, without having to change routing protocols or reassign existing network addresses. Finally, the VPN facilities provided by NSG are the key to NetRanger's secure communication system.

Performance Capabilities

NSG packet filter devices fall into three distinct price/performance categories: the BorderGuard 1000, BorderGuard 2000, and the Passport. These packet filter devices serve as the basis for the three NSX configurations currently offered by WheelGroup: the NSX 1000, 2000, and 5000 systems. As with the NSG systems, the primary difference between the NSX systems is one of network performance, which is summarized in Table 1.1. For detailed hardware information on any of these systems, refer to Appendix E in this *User's Guide*.

Table 1.1: NSX Configurations

	NSX 1000	NSX 2000	NSX 5000
Maximum Bandwidth	512 Kbps	T1 (or 10 Mbps)	T3 (or 100 Mbps)
NSG Device	BorderGuard 1000 • 2 Ethernet • 1 WAN/1 Ethernet	BorderGuard 2000 • 4 Ethernet • 2 Ethernet/2 WAN	Passport • FDDI **(router mode only)** • Ethernet • WAN • Token Ring **(router mode only)**
NSX Sensor	Pentium 166 MHz • 2 GB Hard Drive • 32 MB RAM • 10BaseT Ethernet • modem dial-up • 2 serial ports	Pentium 166 MHz • 2 GB Hard Drive • 32 MB RAM • 10BaseT Ethernet • modem dial-up • 2 serial ports	UltraSPARC 170 MHz • 4 GB Hard Drive • 64 MB RAM (min) • SBUS FDDI (optional) • 100/10BaseT (optional) • modem dial-up • 2 serial ports

NetSentry

As noted earlier, NSX intrusion detection is based on the **monitoring of an open network connection rather than a closed one.** The NSX does this by leveraging the layered filter architecture built into NetSentry, which is diagrammed in Figure 1.3.

The *First*, *Apply Table*, and *Last* filter points apply to **all** of the network interfaces installed on a BorderGuard/Passport system, whereas the *Incoming* and *Outgoing* filter points allow different **in** and **out** policies to be applied to each of the network interfaces installed on the device.

One of the reasons the NSX system is able to perform **real-time** intrusion detection is because it is able to leverage the **copy_to/log_to** auditing features via filters applied to the *First* filter point. For example, the NSX's default *first.fil* filter instructs the BorderGuard/Passport to only pass on specific IP packets to *sensord*. This helps to dramatically reduce the amount of traffic the NSX system has to process.

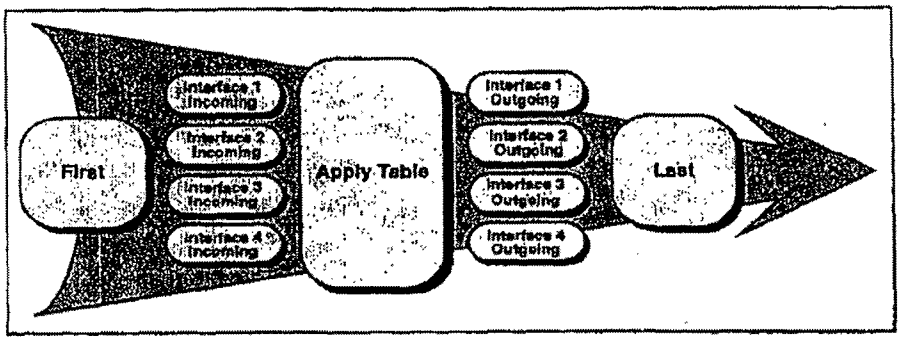


Figure 1.3: NetSentry PCF Filters.

Attack Signatures

As previously mentioned, an attack signature is a *pattern of misuse* based on one or more events. Such a pattern can be as simple as an attempt to access a specific port on a specific host, or as complex as sequences of operations distributed across multiple hosts over an arbitrary period of time. Events can be grouped into different *attack signatures*. An event that is based on a single ICMP packet at a specific point in time is an **atomic signature** (e.g., a ping of a specific host). **Composite signatures**, on the other hand, are based on series of events. A ping sweep is an example of a signature that spans a network; a port sweep is an example of a signature that focuses on a specific host. A SATAN attack is an example of a composite signature derived from a host port sweep pattern. Many of these patterns are also *sequence-independent*, which means that the attack signature must be identified regardless of the order or the duration between atomic events. Other signatures, such as SYN attacks, are based on well-defined event sequences.

Attack Responses

The NSX system does one or more of the following things once an attack has been positively identified:

- Generate an alarm.
- Shun the attack.
- Log the alarm event.

In keeping with the flexibility of NetRanger, the initiation of these actions, as well as how they are initiated, is highly configurable.

Alarms

Alarms are generated by the *sensord* daemon, and are typically routed to a remote Director system. These notifications can also be routed to **multiple** Director systems.

The type of alarm generated for a specific attack is configurable. Alarm notifications are grouped according to their **severity of misuse.** You can define up to 255 different levels of severity. NetRanger's default configuration currently specifies 6 levels of misuse.

Shunning

In addition to generating alarms, *sensord* can initiate **optional actions.** The most common action is to **shun an attack,** which typically involves reconfiguration and reloading of the NetSentry filters. This type of automated response should only be configured for attack signatures with a low probability of *false positive* detection. A SATAN attack is an example of an unambiguous activity, whereas a content-based signature such as VRFY (off mail port 25) is more prone to a false positive pattern match.

Another way of shunning patterns of misuse is to manually reconfigure the BorderGuard or Passport device through the Director system. Shunning is part of a site's security policy that must be carefully reviewed before it is deployed, whether as a set of automatic rules or as a set of guidelines upon which operational staff rely.

Logging

All NSX log data is written to **flat files,** which can then be exported to industrial-grade databases by the NetRanger SAP subsystem described in the *Director* section. Data is written directly to flat files instead of a database in order to maximize **fault tolerance** and **performance.**

The NSX system currently supports two types of logging:

- Event logs
- IP Session logs

Alarms represent just one type of *event* that can be logged by the NSX system. Event logs (examples of which are shown in Figure 1.4) can also contain entries for every *command* and *error* that is generated by a user or NetRanger service. Data is written to these types of log files as long as NetRanger is running. Log files are **serialized** based on configurable time and size *thresholds.* The naming convention for event log files is **log.<date-time>.**

IP Session logs are either continually active or are only written to when a certain event(s) occurs, such as a connection request from a specific IP address, or detection of a string such as "Confidential." When these types of conditions are met, *sensord* can be configured to write every *incoming* and *outgoing* packet to an IP Session log for a predefined period of time. IP Session logs allow you to reconstruct the *conversation* that takes place between a source and destination IP addresses by using the transcript program. The naming convention for IP Session logs is **iplog.<src IP address>**. An example of an IP Session log is illustrated in Table 1.2.

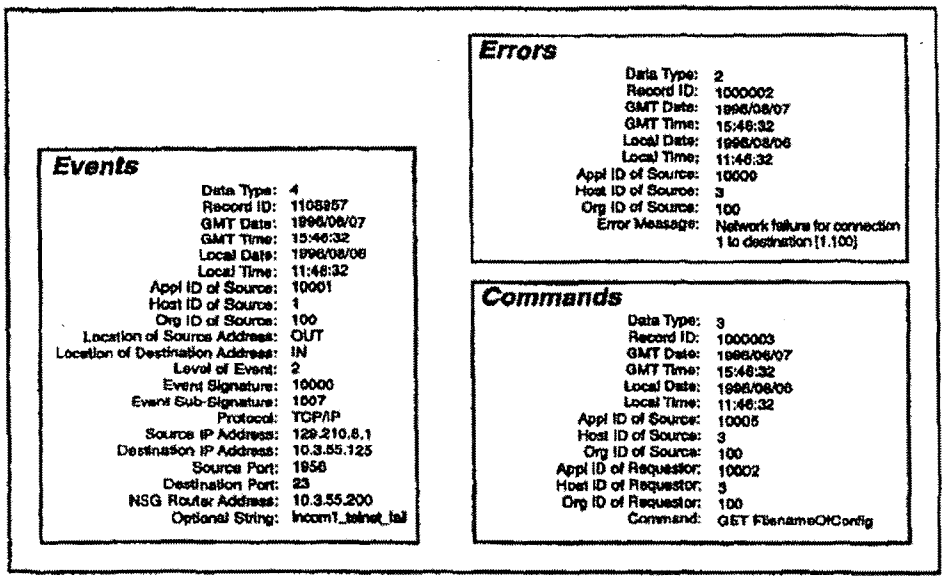


Figure 1.4: Event Log Formats

Table 1.2: IP Session Log Format

Timestamp	Packet Length	IP Packet
4 bytes	4 bytes	20+ bytes

Note that both Event and IP Session information can be logged locally on an NSX system as well as remotely on Director systems; exactly what information is sent where depends on how each NSX system is configured.

Communications Protocol

As Figure 1.5 illustrates, all of the NetRanger services communicate with one another via *postofficed* daemons. Note that this holds true for communication between services on the same host as well as across hosts. All communication is based on a unique three-part address that includes **Organization, Host,** and **Application** identifiers, which are defined in each NSX's and Director's configuration files.

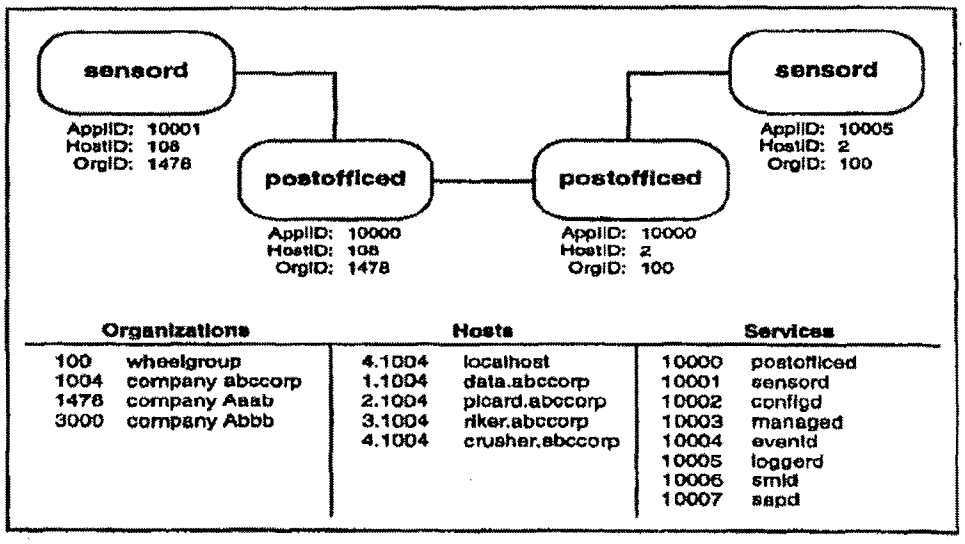


Figure 1.5: Example /usr/nr/etc/ communication files

The benefits of this proprietary addressing scheme are twofold: 1) it can be layered on top of existing network protocols and 2) it can address a much larger domain than the current 32-bit IP protocol.

Alternate Routes

NetRanger's three-part addressing scheme serves as the basis for a point-to-point protocol that allows for up to 255 alternate routes between two hosts. Table 1.3 shows two alternate routes to the "abccorp" host "data". The preferred path is via an IP host with the address of 10.3.55.151; an alternate path has been identified via a host with an IP address of 10.1.4.198.

Table 1.3: Example /usr/nr/etc/routes file

data.abccorp	1	10.3.55.151	45000	1
data.abccorp	2	10.1.4.198	45000	1
picard.abccorp	1	10.1.4.198	45000	1

An important feature of this alternate routing protocol is that it automatically switches to the next route whenever the current route fails. It also uses a system heartbeat to detect when a connection to the preferred route can be reestablished. A system error message is generated (and logged) whenever a connection goes down, and any packets that were lost during the state transition are resent.

Distribution of Data

In addition to specifying alternate routes and maintaining fault tolerance, the communication protocol allows you to define **what** types of information and **how much** information should be sent to each destination.

As noted earlier, the types of information generated by the NSX daemon falls into four basic categories: **Events, Commands, Errors,** and **IP Packets.** The /usr/nr/etc/destinations file allows you to specify what types of information should be routed to which **daemons** on which **hosts.** Table 1.4 shows two different distribution entries. The first entry routes all of the standard Event data to the *loggerd* service on a Director machine named crusher, and the second entry specifies that only *events* should be displayed by *smid* on the riker Director machine.

Table 1.4: Example /usr/nr/etc/destinations file

1	crusher.wheelgroup	loggerd	1	Events, Errors, Commands
2	riker.wheelgroup	smid	2	Events

In addition to specifying what *types* of information should be distributed, NetRanger can dictate what *levels* of events should be sent to each destination. As shown in Table 1.4, only events of level "2" *and above* are being sent to *smid* on *riker.*

Distribution Hierarchies

Another feature that complements alternate routing is the ability to build **hierarchies** of NSX and Director systems through the use of **message propagation.** Instead of broadcasting events from an NSX onto multiple hosts, information can be sent to a single host, which can then propagate packets onto other platforms defined in its local configuration files. Figure 1.6 illustrates this concept via a simple hierarchy of Director machines.

In addition to providing performance benefits and fault tolerance, distribution hierarchies can simplify system management. For example, local Director machines might be responsible for monitoring from 9AM to 5PM and then transfer control onto a central Director every evening.

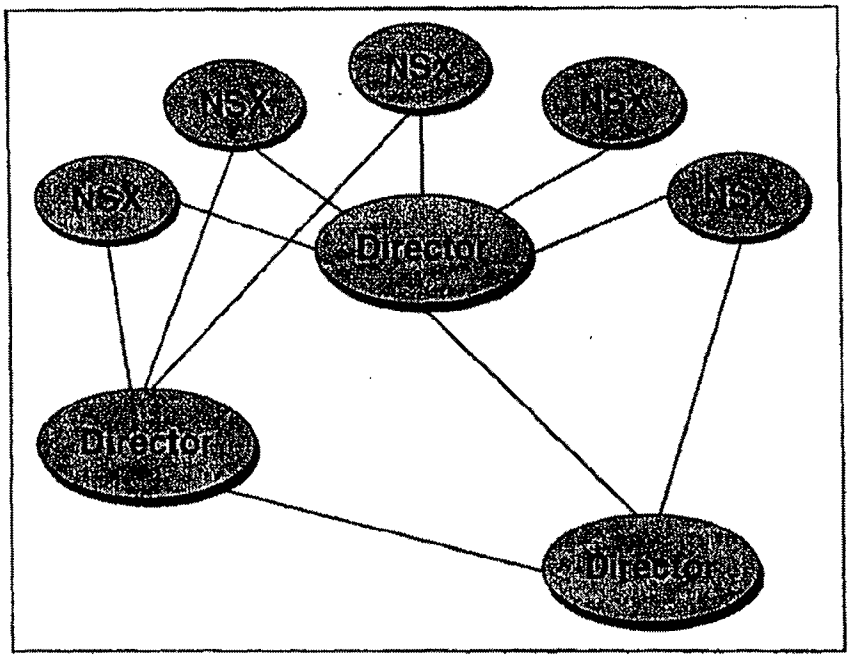


Figure 1.6: Director Hierarchy Based on Message Propagation

Encrypted Sleeves

A key requirement of the communication architecture is its **security**. This requirement is currently met by passing all communication between NSX and Director systems through NSG's Data Privacy Facility (DPF), which is illustrated in Figure 1.7. The DPF maintains Virtual Private Networks **(VPN)** via **RSA** public key and one of the following private key systems: **DES**, **Triple DES**, or **IDEA**. The DPF is noteworthy in that

- it can be maintained across *collections* of NSX and Director systems, and
- it is *transparent* to applications such as NetRanger.

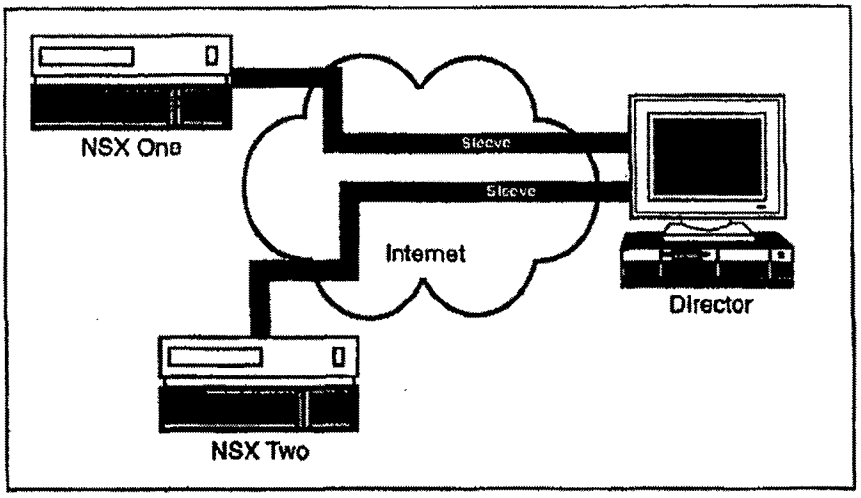


Figure 1.7: Example of NSG's VPN

Director System

As noted earlier, the NetRanger Director consists of two major subsystems:

- Security Management Interface (SMI)
- Security Analysis Package (SAP)

These two subsystems provide **centralized command and control** of an organization's security perimeter, which could conceivably encompass hundreds of NSX systems. From a capabilities perspective, these two subsystems provide **monitoring, management, data collection, data analysis,** and **user-defined actions** services. All of these capabilities are supported by the *smid, configd, loggerd, sapd,* and *eventd* daemons diagrammed in Figure 1.2. There is also an application called *nrdlmap* that serves as the interface between *smid* and HP OpenView.

NSX Monitoring

The Director's most prominent capability is display of **real-time** event information, which is based on the *smid* daemon, the *nrdirmap* application, and OpenView. The *smid* daemon accepts incoming event records from one or more NSX systems via a local *postofficed* and translates them into a format that *nrdirmap* understands. The *nrdirmap* application uses the OVW API (OpenView Windows Application Programming Interface) to tell the OpenView user interface what security information to present to the user. Security information is presented via **icons** drawn on one or more **network security maps**. Figure 1.8 shows an example of the Director's display of a collection of NSX maps.

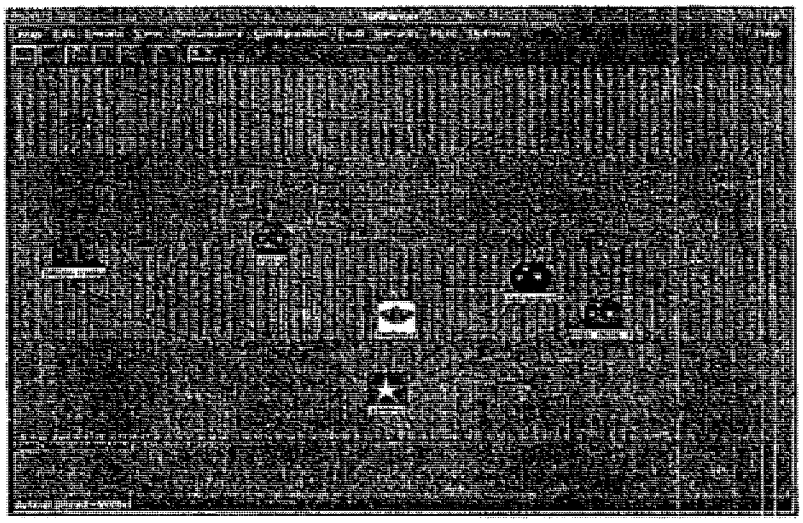


Figure 1.8: Example Director Network Security Submap

Network Security Maps

The Director arranges icons into hierarchical security maps based on OpenView's Network Node Management (NNM) user interface. If you double-click an icon, you can view the next lower "submap." Figure 1.9 shows several examples of *Event/Alarm* icons. The Director hierarchy includes the following levels, where *NetRanger* is the "root" and *Events/Alarms* are the "leaf" nodes of the tree:

- NetRanger
  - Collections of Directors/NSXs
    - A Single Director or NSX System
      - Applications running on a Director or NSX System
        - Events/Alarms generated by an Application

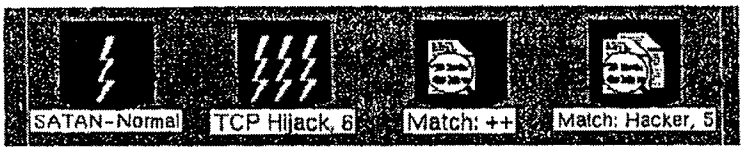


Figure 1.9: Sample Director Alarm Icons

Icon States

Every icon within a network security map has a **state**, which is expressed in the form of textual and graphical attributes.

The most visible indication of an icon's state is its **color**. The colors **green, yellow**, and **red** represent the states **normal, marginal,** and **critical**. These states are user-defined in an attribute dialog (shown in Figure 1.10). Level 2 and 3 alarms are usually set as *marginal* (yellow), and level 4 and 5 alarms are set as *critical* (red).

Unlike all other icon attributes, color is a state that is **inherited** from *Event/Alarm* icons by all other levels of the icon hierarchy. For example, an NSX machine icon is automatically set to red if any of its subordinate NSX applications has received a critical alarm. The **ability to propagate** alarm states up through a hierarchy is one of the features that makes the Director such a powerful network security monitoring tool.

Every icon in a Director hierarchy also possesses textual attributes. This information is accessed by selecting a particular icon and then choosing **Describe→Modify** from the menu bar or pressing **Ctrl+O**, which brings the icon's attributes dialog forward. Each type of icon has a different set of attributes. Figure 1.10 shows the attributes for an NSX *sensord* icon.

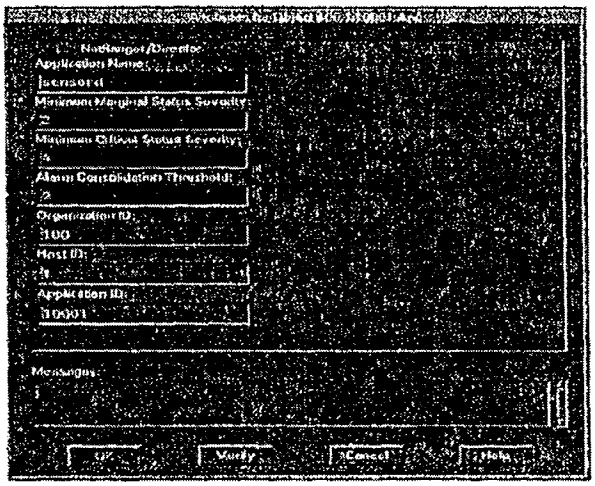


Figure 1.10: sensord Attribute Information

NSX Management

Another key capability of the Director is remote management of the services (or *Applications*) that make up an NSX system. *Applications* are managed via a graphical user interface called **nrConfigure**, which talks directly to the *configd* daemon. This interface allows a user to effectively *get* and *set* such attributes as log file names and alarm responses. The nrConfigure interface is activated by selecting one or more icons on the security map and then choosing **Security→Configure** from the OpenView menu. This opens the window shown in Figure 1.11.

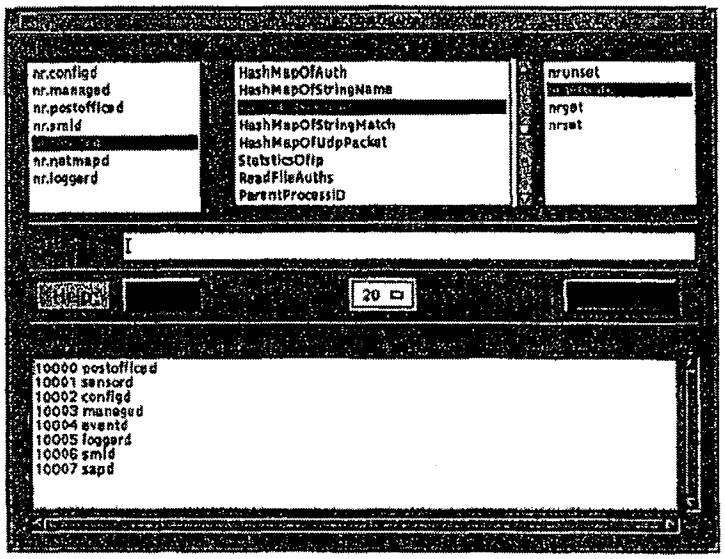


Figure 1.11: NSX Configuration Interface

The **Applications** displayed in nrConfigure's top left-hand list box represent the services running on the first NSX icon highlighted by the user. The top center list box displays all of the **Tokens** that apply to the Application currently highlighted in the left-hand list box. The top right-hand list box displays the **Actions** that are allowed for the currently selected Token. Figure 1.11 shows the Actions and Tokens for the *sensord* daemon.

NetRanger currently supports over 100 tokens. An *Action* is put into effect by pressing the **Execute** button. The five possible actions are **get, getbulk, set, unset**, and **exec**.

The **get** and **getbulk** commands are read-only operations that obtain information from an *Application*. The **get** command returns a *single* item of information; **getbulk** returns a *set* of items. Figure 1.11 shows the results from a getbulk request for all of the services defined for an NSX or Director.

The **set** command is a write operation that updates the value of a *Token*; **unset** removes a specific Token and its value from an Application's configuration. Adding or deleting a content-based attack signature (such as VRFY) from an NSX is one example of these *Actions*.

The **exec** command instructs an application to execute an action external to itself. For example, you can instruct an NSX to shun a specific IP address by issuing an **exec** against *managed*, which in turn modifies the filter configuration on its packet filter. The **exec** command is also used for such tasks as writing configuration information to disk.

It is important to understand that all configuration data is read and written to or from the *applications* themselves, not from local caches or parameter files. The results of a **set** or **exec** take effect immediately on the target NSX.

Finally, all of the functionality provided by nrConfigure is also available via command-line interfaces. This allows trusted users without access to a Director to manage NSX systems from a simple terminal session.

NSX Data Collection

NSX data collection serves as the foundation for the SAP subsystem, and is based on the *loggerd* and *sapd* services diagrammed in Figure 1.12. These daemons use a simple **push-pull** mechanism to migrate data into a remote database. As explained earlier, *loggerd* pushes data into flat files, which are serialized based on configurable size or time thresholds. This data is then pulled into a remote database by *sapd*, which has its own polling interval.

Writing to intermediate flat files in this manner provides levels of **fault tolerance** and **performance** cannot be achieved when writing directly to a database. Data throughput in a distributed application such as NetRanger is constrained by the weakest link in the system.

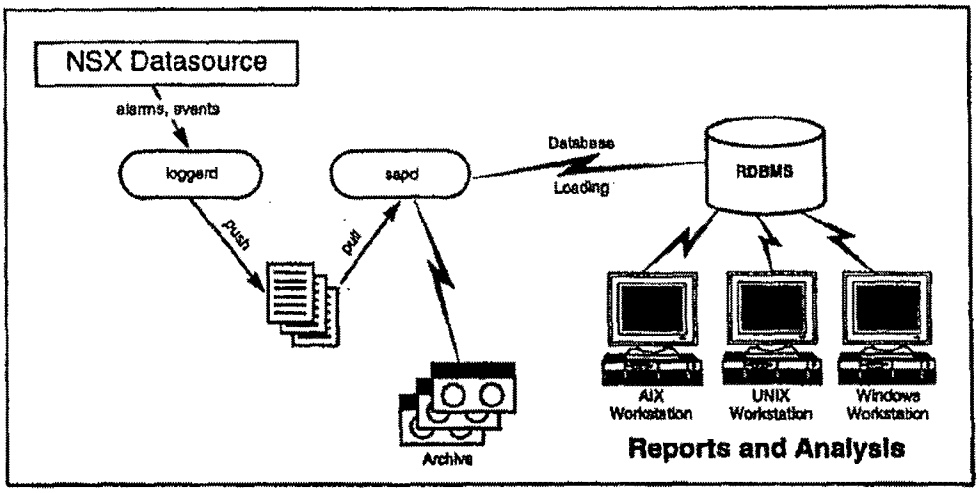


Figure 1.12: NSX Data Collection

With the SAP system, the data capture process is insensitive to database availability or performance fluctuations.

The SAP currently ships with *sapd* drivers for **Oracle®** and **Remedy®**. However, SAP can also be configured to write to other databases, such as Sybase® and Informix®. Example scripts are shipped with the Director that show how a database's native bulk load tools can be easily integrated into *sapd*.

SAP also includes a configurable file management capability that automatically archives event and IP logs, either on an NSX or a Director. Sample file management token profiles are provided for both NSX and Director configurations.

NSX Data Analysis

The SAP capability also analyzes data. Rather than locking the user into a single tool on the Director machine, this task is better served by **3rd-party tools** on a separate Windows platform, such as the IQ Objects® **report writer** from IQ Software and **multi-dimension analysis** tools such as PowerPlay® from Cognos. **Trouble ticketing** systems such as Remedy's Action Request System® (ARS) can also be implemented on top of NetRanger's alarm data.

As a foundation for custom reports, SAP includes a subsystem called SAPR. This is a set of SQL queries which can be easily customized or integrated into the 3rd-party tools.

These types of 3rd-party tools can be configured to support ad hoc queries as well as predefined reports. For example, these tools can easily generate reports showing

- all alarms of levels 4 and 5 in the last 30 days,
- a graph of Web server activity over the last 24 hours, and
- a table of all events in the last 30 days in order of increasing alarm levels.

User-Defined Actions

In addition to displaying and logging alarm events, the Director can generate user-defined actions via *eventd*. A typical action might be to generate pager notifications via e-mail messages or feed data onto 3rd-party devices, such as a printer. Support for multiple action scripts is also provided. While *eventd* makes no distinction between alarm types and levels, the default action script shipped with this service shows how actions can be triggered based on these criteria.

NETRANGER PRE-INSTALLATION 2

What to do Before Installing NetRanger 1.3.1

Before installing NetRanger, it is important to have a complete understanding of the current corporate network architecture. This is a critical step in the pre-installation process. Failure to obtain proper information could result in the creation of network security holes and loss of network functionality and services. This section describes what steps to take before installing the system on a network.

Analyze the Current Network Architecture

The first step in protecting a network is understanding the existing network architecture and requirements. The most effective way to analyze a network's current architecture is to follow these steps:

1. **Identify what to protect.**
2. **Define all entry and exit points to the protected network.**
3. **Identify current security measures.**

Think about which network assets need protection. Also, closely examine the connectivity *between* corporate networks that might give unwanted access to the network environment. If the configuration contains connections between more than one physical site, determine whether to protect the remote assets. If not, then determine whether to protect the network *from* the remote site.

Define All Entry and Exit Points to the Protected Network

This is the most important, and possibly the most difficult, step in protecting a network. The way an organization is connected physically as well as electronically affects identification of all network entry and exit points. Some typical scenarios follow.

Case 1—One Geographic Location, No Existing Internet Connection

This is the simplest case. If NetRanger is being installed at the same time an Internet connection is being established, then the NetRanger NSX can be installed as your Internet router as well as an intrusion detection and response system.

In order to isolate your internal network from other corporate networks as well as the Internet, identify all routers local to that network and either replace them with a NetRanger NSX or place a NetRanger NSX behind them. Refer to the *BorderGuard Setup* and *BorderGuard Configuration* sections in this *User's Guide* for detailed instructions on how to do this.

If your network connects to one or more business partner networks, security can be enhanced by placing a NetRanger NSX on that connection. It is difficult to control how a business partner's network is configured or what security countermeasures they have put in place to deter threats.

Case 2—One Geographic Location, Existing Internet Connection

In this situation, your organization has only one physical site, but more than one Internet connection. Each Internet connection must first be identified by contacting your Internet Service Provider to verify connectivity and registered Internet address ranges. Each registered address range assigned to your company should then be accounted for in the existing Internet router.

Refer to Case 1 for instructions on how to further secure your network.

Case 3—Multiple Geographic Locations, Existing Internet Connection

This is the most complicated situation. It is very likely that each site has its own Internet connection as well as a path to the network that needs protection. You have three options:

- **Protect your network at the Internet connection and at the connections to other sites.** This will secure all entry and exit points to your protected network. However, the remote sites will remain unprotected and the protected network will be more directly accessible to potential intruders.
- **Protect your local network and all remote network sites at their connections to the Internet.** This protects the corporate network from Internet intruders, but it does not secure communication between the corporate network and remote sites or business partners connected to your network.

- **Place NetRanger on all connections to the Internet, at connections to remote sites and at connections to business partners.** This provides the highest level of security and protects your network from internal as well as external (Internet) attacks.

Please note that the last two options require you to closely coordinate with network administrators at the remote sites to ensure complete coverage.

Identify Current Security Measures

How Existing Firewalls Affect NetRanger Installation

The NetRanger NSX can work in conjunction with existing firewalls as long as they are appropriately placed within your network. Ideally, the NetRanger NSX should be located *in front of* the existing firewall. If the NetRanger NSX packet filter is placed behind a firewall, intrusion detection will be limited to traffic that has been allowed through the firewall, and it will not generate alarms on reconnaissance or failed malicious activity. In cases where there is no need for address translation or other proxy services, NetRanger can replace firewalls. If there is no need for address translation or other proxy services, NetRanger works best without a firewall and has less impact on network performance.

How Security Filters in Existing Routers Impact NetRanger Installation

Please note that filters on existing routers must be removed for NetRanger to function **at its optimum level**. NetRanger will provide the same filtering capability in addition to its robust intrusion detection and response capabilities.

NSG BorderGuard Installation Options

Once your network architecture has been analyzed and you have identified your current security measures, the next step is to decide where to place the BorderGuard security device on the network. If you already have an established network, the simplest and least intrusive way to install the BorderGuard is as a **bridge device**. Otherwise, the simplest installation is to install the BorderGuard as a **router**. This section explains these as well as four other installation options. Please refer to the *BorderGuard Configuration* section for specific information about configuring the BorderGuard.

Option 1—Install the BorderGuard as a Bridge

You will typically want to deploy the BorderGuard as a bridge when the network contains existing routers. In bridge mode, the BorderGuard can be placed in front of or behind your existing Internet router. It is usually simpler to place the BorderGuard on the LAN side. No subnetting of existing networks or additional networks is necessary. In most cases, the BorderGuard requires a single host address for configuration and VPN sleeves. Please

note that when an NSX Sensor and Director are directly connected to the BorderGuard (or they use an out-of-band channel), no IP address is needed. Thus NetRanger can be also installed in situations where there are *no* available IP addresses. The basic configuration for this installation option is diagrammed in Figure 2.1.

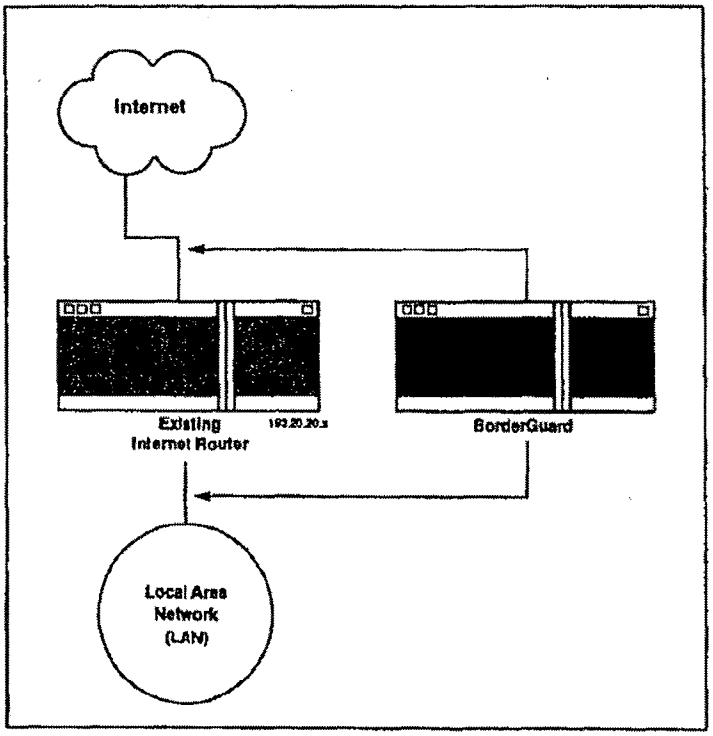


Figure 2.1: BorderGuard Router in Bridge Mode

Option 2—Install the BorderGuard as the Internet Router

If your network has not yet been connected to the Internet, the simplest way to install the BorderGuard is as your Internet router. A BorderGuard 1000 or 2000 is usually connected to a Local Area Network (LAN) via Ethernet port(s). Connections to a CSU/DSU depend on the type of unit. Most CSU/DSU units have a V.35 serial connection; a small percentage use RS232 or RS449.

Your Internet Service Provider (ISP) assigns the external IP address, but all internal IP address(es) also need to be available for BorderGuard installation. If a connection to the Internet already exists, the current Internet router should be replaced and its IP address should be applied to the BorderGuard's external Ethernet. The basic configuration for this installation option is diagrammed in Figure 2.2.

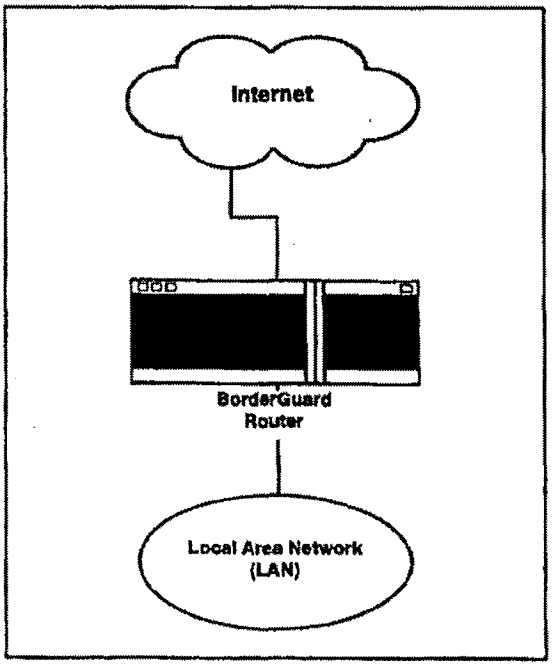


Figure 2.2: The BorderGuard as the Internet Router

Options 3–6—Unable to Replace the Existing Internet Router

The remaining configuration options focus on situations where it is not possible or feasible to replace an existing Internet router with a BorderGuard unit (for example, an ISP may require a particular brand of router, such as a Cisco®) and bridging is not desired. The following situations are discussed. You have

- a class B address,
- one or more unused class C addresses,
- no unused class C addresses, or
- a class C address that cannot be subnetted.

Option 3—A Class B Address

In this case, your Internet connection is based on a class B Internet address (the first number in the IP address is between 128 and 191, such as 130.130.x.x). An unused logical class C address can be assigned to the interface between the current router and the BorderGuard (such as 130.130.10.x). All current addresses can remain the same, and there is a minimal amount of overhead. This configuration is diagrammed in Figure 2.3.

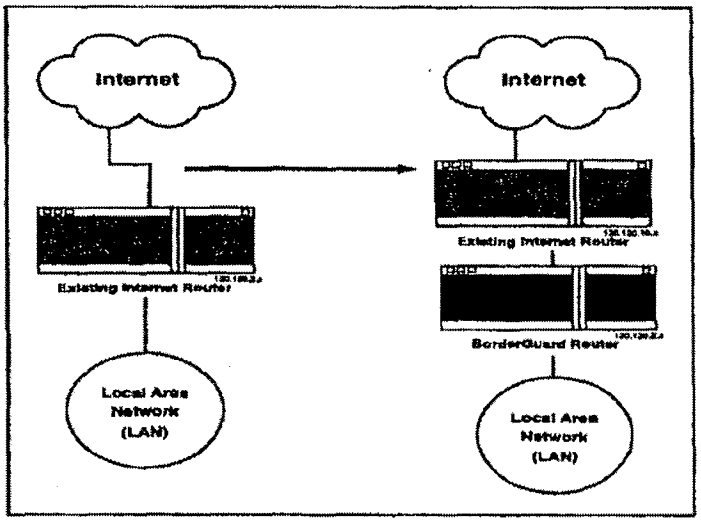


Figure 2.3: BorderGuard Router Placed Behind the Existing Internet Router (Class B Address Space)

Option 4—An Unused Class C Address

Another option when the existing router cannot be replaced is to assign an unused class C address (the first number is greater than 191, such as 193.20.20.x) between the existing router and the BorderGuard. This can be done if multiple class C addresses exist and at least one of them is currently unused. As with Option 3, the remaining IP addresses can remain the same. This configuration is diagrammed in Figure 2.4.

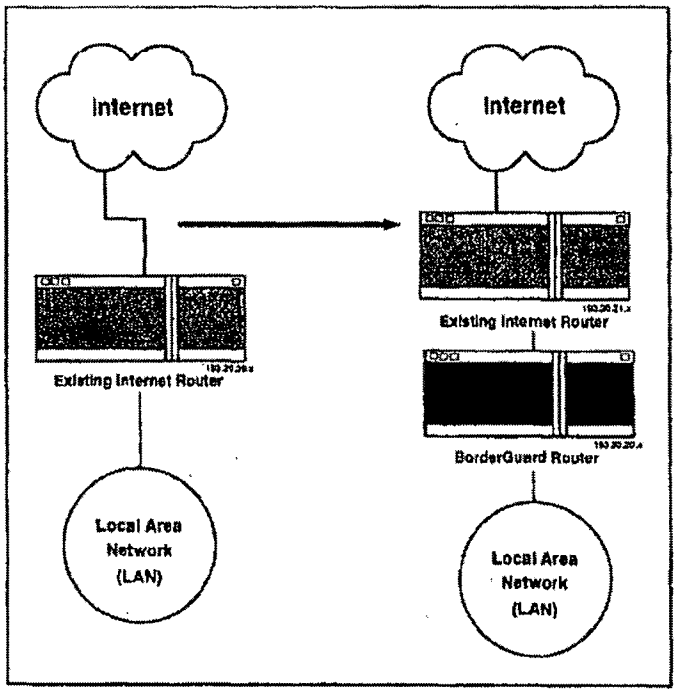


Figure 2.4: BorderGuard Router Placed Behind the Existing Internet Router (Multiple Class C Addresses)

Option 5—Unable to Replace Existing Internet Router and No Unused Class C Addresses

In some cases, it may not be possible to replace an existing Internet router and there are no unused class C addresses (the first number is greater than 191, such as 193.20.20.x). In this situation, subnet an existing class C network and change some internal host addresses. **Always try to subnet the class C address with the least number of hosts.** The addresses that must be changed are those that end in the first subnet range, such as x.x.x.1-30 for netmask 255.255.255.224, or those that end in the last subnet range, such as x.x.x.225-254 for netmask 255.255.255.224. This configuration is diagrammed in Figure 2.5.

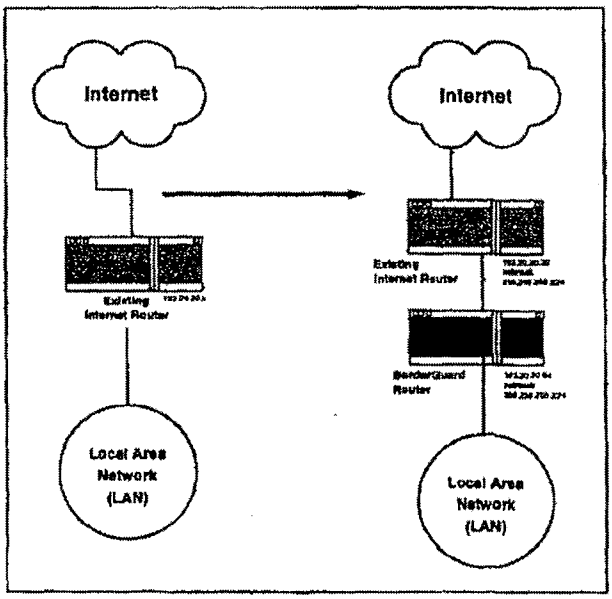


Figure 2.5: BorderGuard Router Placed Behind the Existing Internet Router (Class C Address is Subnetted)

Option 6—Unable to Subnet The Existing Class C Address

In some cases, it is impossible to subnet a current class C address. This typically happens when only one registered class C address exists and all of its addresses are assigned. There are two options for installing the BorderGuard in this situation:

- Obtain another class C address from the InterNic and employ Option 3 described earlier.
- Purchase a proxy server or address translation device and remap the internal IP addresses.

The second option also requires that subnetting the class C address assigned to the BorderGuard router and the proxy server. While both of these options are fairly difficult, obtaining a new address is the least problematic of the two. These options are diagrammed in Figure 2.6.

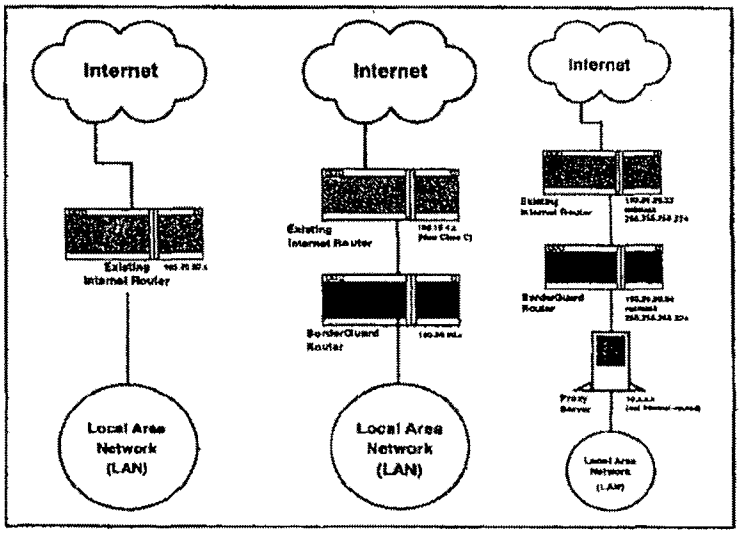


Figure 2.6: BorderGuard Router Placed Behind the Existing Internet Router

NetRanger NSX Sensor Installation Options

In addition to determining how to deploy the BorderGuard, you will also need to consider the various options you have for placing the NSX Sensor on your network. Each option carries with it different costs, such as extra ports on the BorderGuard; extra hardware (switched Ethernet hubs); and different benefits, such as increased security and performance. You can install the NSX Sensor on your network in one of the following ways:

- Install the NSX Sensor on a separate, isolated network.
- Install the NSX Sensor on the corporate network.
- Install the NSX Sensor on a switched Ethernet network.

Option 1—Install the NSX Sensor on a Separate, Isolated Network

The most secure NSX configuration is when the NetRanger NSX Sensor is placed on its own network. This configuration can only be implemented with the NSX 2000 or 5000. Use one of the Ethernet interfaces on the BorderGuard 2000 to create a private network for the NSX Sensor. The benefit of this configuration is that the traffic traveling between the BorderGuard and the NSX Sensor can be protected by the BorderGuard's security policy. This configuration also performs slightly better since communication between the BorderGuard and the NSX does not have to pass across your corporate network along with other traffic and is similar to the diagram illustrated in Figure 2.7.

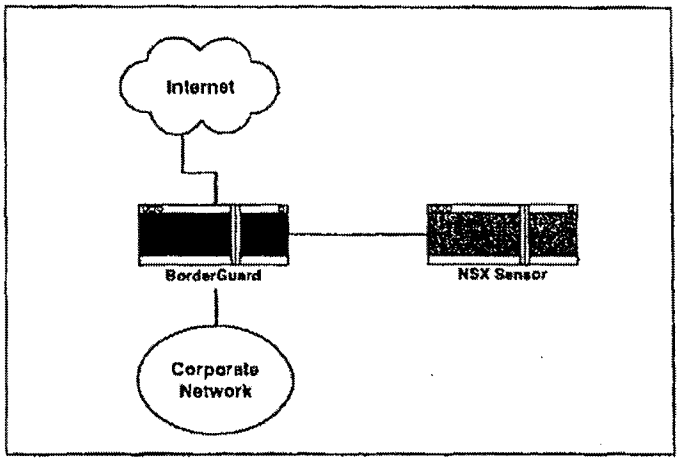


Figure 2.7: The NSX Sensor Placed on its Own Isolated Network

Option 2—Install the NSX Sensor on the Corporate Network

In this case, the NetRanger NSX Sensor resides on the internal network. This configuration can be used with either the NSX 1000, 2000, or 5000. This configuration (which is diagrammed in Figure 2.8) has two disadvantages:

- Communication between the BorderGuard and NSX Sensor devices is mixed in with existing internal traffic and may create some overhead on the internal network.
- The NSX Sensor is exposed to attacks from systems within the internal network. Unlike the first option, the BorderGuard filters cannot be used to protect the NSX Sensor from these types of attacks.

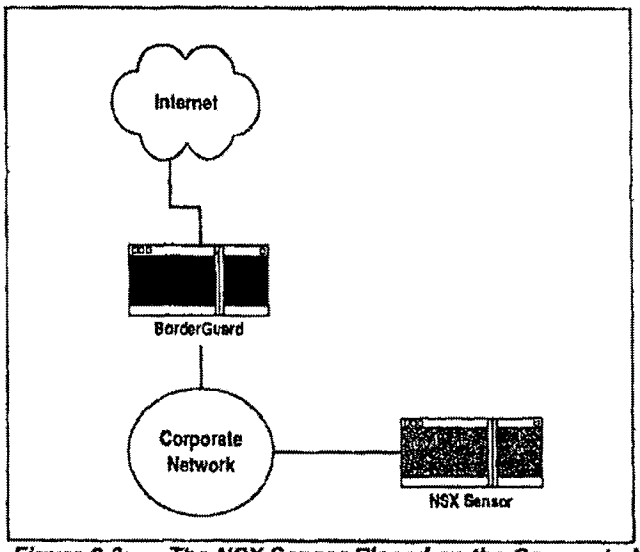


Figure 2.8: The NSX Sensor Placed on the Corporate Network

Option 3—Install the NSX Sensor on a Switched Ethernet Network

With this option, the NetRanger NSX sensor resides on the same network as the rest of the users. However, the NSX Sensor is isolated from those users with a switched Ethernet hub. This configuration offers the same performance advantages as attaching the NSX Sensor directly to the BorderGuard, and it also provides additional security from the corporate network. The security won't be as robust as with Option 1, but the NSX Sensor traffic is protected from users on the internal network, which does provide some additional security. This configuration is diagrammed in Figure 2.9.

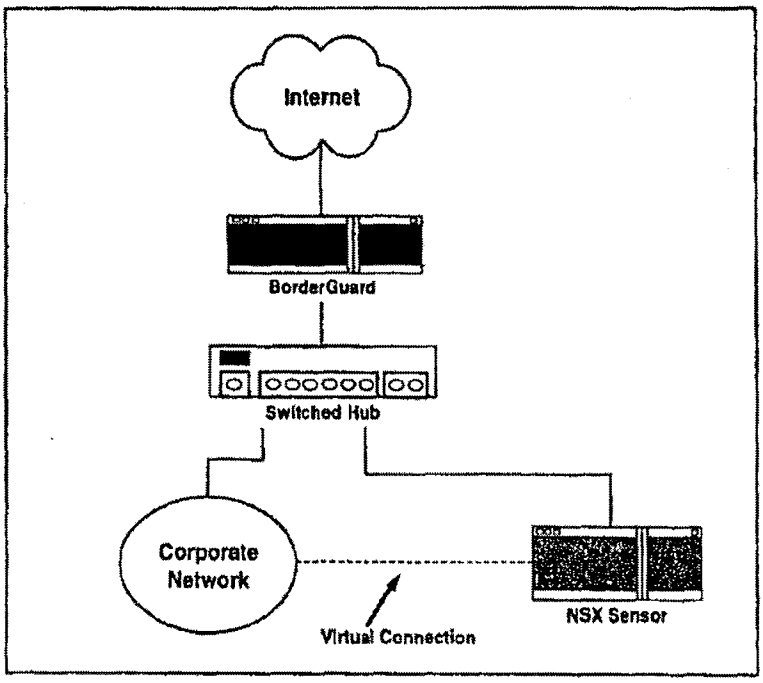


Figure 2.9: The NSX Sensor Placed on a Switched Ethernet Network

NetRanger Director Setup Options

Two things must be considered when deciding where to place the NetRanger Director on a network. The first concerns operational use of the system. The Director should be placed in a physical location that is secure and close to the individual(s) responsible for monitoring the networks. If X-sessions will be used to access the Director, then it should be placed on the same network as the operators of that system. The second consideration relates to the Director's accessibility to NSX systems. There must be a path between the NSX and the Director for the alarm and management functions to work properly. If the Director is going to be placed on a network behind a proxy firewall or an address translation device, a NetRanger post office daemon must be loaded onto that device for communication to take place. Contact a WheelGroup representative for details on supported firewalls and operating systems.

Physical Installation Considerations

Because NetRanger is a significant component of the overall security environment, the NetRanger system(s) should be placed in a secure location. NSX and BorderGuard systems are shipped with rack mounts and can be located near other network equipment.

Power

Both the NSX and Director systems are currently UNIX-based and should be connected to an Uninterruptable Power Supply (UPS) to protect them from power outages and surges.

Physical and Operational Specifications

The BorderGuard 1000

Dimensions

- 1.74" high
- 19" wide (including rack mount)
- 12"" deep
- weighs 8.2 pounds

Environment

- Temperature Range—32° to 113° F
- Humidity Range—5% to 95% relative humidity (noncondensing)

Power Requirements

- Power supply voltage—90-250 VAC
- Power Supply Current—0.3 Amps (110 VAC); 0.15 Amps (230 VAC)
- Power supply frequency—47-63 Hz

The BorderGuard 1000 has an AC self-regulating power supply that will automatically adjust to the supplied voltage.

Regulations (U.S.)

- UL1950, 1st edition

EMI/RFI

- FCC CFR 47 Part 15, Level B

The BorderGuard 2000

Dimensions

- 6.5" high
- 19" wide (including rack mount)
- 17.5" deep
- weighs 25 pounds

Environment

- Temperature Range—41° to 110° F
- Humidity Range—5% to 95% relative humidity (noncondensing)

Power Requirements

- Power supply voltage—90-264 VAC
- Power Supply Current—1.5 Amps (110 VAC); 0.75 Amps (230 VAC)
- Power supply frequency—47-63 Hz

The BorderGuard 2000 has either an AC or 48 VDC self-regulating power supply.

Regulations (U.S.)

- UL1950, 1st edition

EMI/RFI

- FCC CFR 47 Part 15, Level B

The NSX Sensor

Dimensions

- 7" high
- 19" wide (including rack mount)
- 20" deep (including front handles)
- weighs 32 pounds

Environment

- Operating Temperature—10° to 30° C
- Storage Temperature— -40° to +70° C

Power Requirements

- AC Voltage/Frequency—115v/60Hz; 230v/50Hz (switchable)
- Power Supply Current—1.5 Amps (110); 0.75 Amps (230)

Regulations (U.S.)

- UL1950, 1st edition

EMI/RFI

- FCC CFR 47 Part 15, Level B

CONFIGURATION AND INSTALLATION 3

Proper installation of the NetRanger system involves the following steps:

- Install and Configure the NetRanger NSX Sensor.
- Install and Configure the BorderGuard or Passport Security Device.
- Install and Configure the NetRanger Director.
- Complete the nrconfig Utility Overview and Worksheets.
- Define the Security Policy.

Install and Configure the NetRanger NSX Sensor

Installing the NSX involves these steps:

- Position the NSX in a secure and stable location.
- Attach power, network, and modem cables.
- Access and configure the NSX system.
- Perform NetRanger-specific configuration.

Position the NSX

WheelGroup recommends that you place the NSX in a secure location and physically close to the BorderGuard with which it will be operating. The NSX should be placed on a solid, flat, well-ventilated surface, such as a desk, shelf, equipment rack, or wiring closet. Ideally, the NSX should be placed within a standard communications rack.

Attach Power, Network, and Modem Cables

Proper installation of the NSX requires that the power cable and network cable are connected. ***The NSX and BorderGuard should also be plugged into an Uninterruptable Power Supply (UPS) to ensure continuous operation during power failures.*** The NSX is pre-configured to operate using the 10BaseT connector on the Ethernet card. Attachment of a modem cable to the internal modem is optional for most installations. However, this option is required if the initial configuration of the NSX will occur over a dial-up connection.

The NSX operating system is Solaris 2.5.1 x86. It is pre-installed on the internal hard disk along with the NetRanger software. ***The power to the NSX should never be turned off without first properly shutting down Solaris. Failure to do so may cause the file system on the NSX to become corrupted with possible loss of data.*** Repeated power-offs of this kind may cause the NSX to not boot properly or not at all. If this occurs, please contact your NetRanger maintenance provider.

Access and Configure the NSX System

You can log in through one of the following methods to configure the NSX:

- network
- modem
- serial
- console

Network Access

You can access the NSX login prompt via the network. When you use this method of access, the login prompt will look like the following:

```
UNIX(r) System V Release 4.0 (nsx)
login:
```

This requires attaching a computer to the same Ethernet LAN as the NSX and then using Telnet to connect to the NSX. The default IP address of the NSX is 10.1.9.201 with a netmask of 255.255.255.0. This means that the computer used to access the NSX must be configured with an IP address between 10.1.9.1 and 10.1.9.254 excluding 10.1.9.201.

Modem Access

You can access the NSX login prompt via the internal modem. When you use this method of access, the login prompt will be

```
modem login:
```

This requires the installation of a standard telephone cable to the RJ-11 jack labeled TELCO located on the back of the NSX. The modem is configured to answer after the first ring. The NSX is pre-configured for a vt100 terminal once the modem connection is established.

Serial Access

You can access the NSX login prompt via a serial connection to the COM1 port. When you use this method of access, the login prompt will be

```
ttya login:
```

This requires that a null-modem cable be attached between the NSX and a vt100-compatible terminal or a computer running terminal emulation software. Set the terminal or terminal emulation software to the following specifications:

```
Terminal:  VT100
Baud Rate:  9600
Word Length:  8 bit
Stop Bit: 1 bit
Parity:  None
```

Console Access

You can access the NSX login prompt via the console. When you use this method of access, the login prompt will be

```
nsx console login:
```

This requires the attachment of a keyboard and monitor to the NSX. A standard VGA-compatible monitor is adequate for access.

Logging In

Log in as user **netrangr** at the NSX login prompt. Once the initial prompt is accessed, use the **su** command to become the root user. The default root password is **attack**. Once the root prompt is accessed, immediately change the root password by using the **passwd** command.

NOTE

Write down the new passwords you have chosen for both netrangr and root and store them in a secure location.

Initial Configuration

Once you have set the account password, you are ready to configure the machine's basic settings via the **sysconfig-nsx** utility. Note that the utility must be run by user root. The **sysconfig-nsx** command will display a menu, illustrated in Figure 3.1 below.

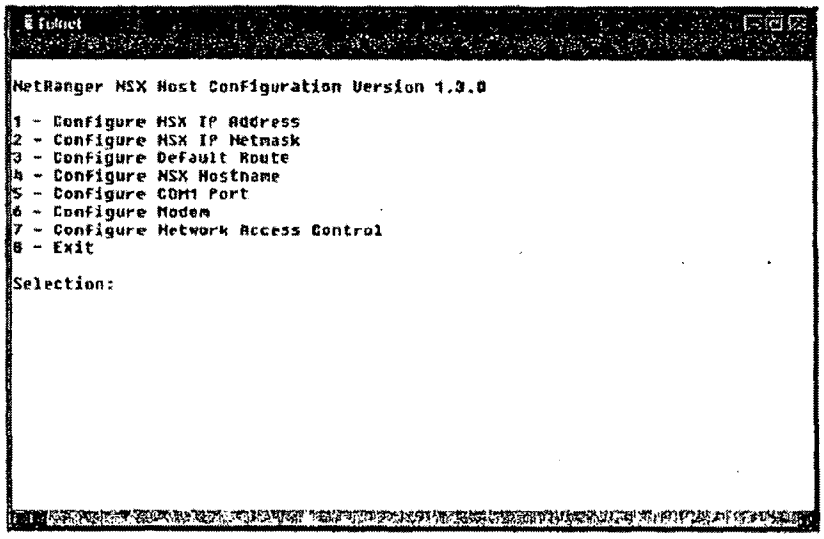


Figure 3.1: NetRanger NSX Host Configuration Screen

Proper installation of the NSX requires configuration of the IP address, IP netmask, and default route. Be sure to configure the network access control to include a list of IP addresses that require Telnet access to the NSX. ***You must include the BorderGuard associated with the NSX in this list because it will require Telnet access to the NSX.*** For the new configuration to take effect, reboot the NSX. After exiting the **sysconfig-nsx** utility, enter the command **init 6** to force the NSX to reboot.

Perform NetRanger-Specific Configuration

You must configure each of the following for each NSX:

- **Communications.** Use the NetRanger Configuration Instructions and Worksheets (immediately following page 3-26).
- **Filters.** Use the Network Security Policy Worksheets (pages 3-28 through 3-35).
- **Alarms.** Refer to Appendix C if you want to alter the default NetRanger alarm settings.

Install and Configure the BorderGuard/Passport Device

BorderGuard Installation

To install the BorderGuard, refer to the NSG BorderGuard Reference manual that corresponds to your unit.

BorderGuard Configuration

The settings required to deploy the BorderGuard are generated for you by the /usr/nr/bin/nrconfig script shipped with the NetRanger NSX Sensor. The nrconfig script places a copy of the BorderGuard configuration files in the /tmp directory of the NSX. The BorderGuard uses a TFTP session to transfer the configuration files from /tmp on the NSX to the BorderGuard. After the files have been transferred and you reboot the BorderGuard, the new settings will take effect.

In the event you need to manually edit any of these settings, please refer to *Appendix B* for manual configuration information. You can also refer to the NSG Reference manual that corresponds to your unit for additional information about these settings.

To generate the NSX and BorderGuard configuration files, follow these steps:

1. **Run the nrconfig utility on the NSX unit as user netrangr.**
   When the nrconfig utility commits the configuration files, it writes the following files to the /tmp directory:
   - filters.cmd
   - first.fil
   - incom1.fil
   - last.fil
   - params.dpf
   - pubkeys.dpf
   - shun.fil
   - sleeves.cmd
   - sleeves.dpf
   - sleeves.fil
   - startup

2. **Change the permissions on the BorderGuard configuration files in the /tmp directory so that all users (user, group, and other) have read access.**

NOTE

The following procedure (steps 1-7, below and on the next page) assumes that the BorderGuard IP address is in the Network Access Control List. To verify or add the BorderGuard IP address to the Network Access Control List, run the **`sysconfig-nsx`** utility as user root and choose menu option 7—**`Configure Network Access Control`**.

To transfer the BorderGuard configuration files from /tmp on the NSX to the BorderGuard, follow these steps:

1. **Log in to the NSX as user netrangr.**

2. **Cable a terminal connection between the NSX and BorderGuard by attaching a direct connect cable to the NSX serial port (COM1) and the BorderGuard console port.**
   For additional information, see the *BorderGuard Cabling and Setup* section.

3. **Type** `tip <borderGuard>` **where** `<borderGuard>` **is either** `bg1000` **or** `bg2000`**, depending on the type of BorderGuard being configured. After the** `connected` **prompt appears on the NSX, press** Enter.

NOTE

The tip command will respond with `all ports busy` if the COM 1 port on the NSX is configured for serial mode and not device mode. To toggle the mode, run `sysconfig-nsx` as the root user and choose menu option 5—`Configure COM1 Port`. Set the COM1 state to `device`.

4. **Type** `ip start if <interface> <bg ip address>` **to assign the selected IP address to the designated interface on the BorderGuard. To verify the network connection,** type `ip ping <nsx ip address>`.

5. **Type** `ip tftp <nsx ip address>` **to initiate the TFTP session with the NSX.**

6. **Type** `get <file>` **for each of the files listed in Step 1.**

7. **After retrieving all the files, type** `quit` **and reboot the BorderGuard.**

BorderGuard Cabling and Setup

Refer to Table 3.1 for the cable and Ethernet transceiver type for your network connections. (The cable and Ethernet transceiver type will vary depending on the network connection.)

Cable Requirements

- The cable required for a BorderGuard 1000-to-NSX connection is a 9-pinM to 9-pinF. (Attach a 9-pinM to the console port on the BorderGuard 1000 and the 9-pinF to the serial port on the NSX.)
- The cable required for the BorderGuard 2000-to-NSX connection is a 25-pinM to 9-pinF. (Attach a 25-pinM to the console port on the BorderGuard 2000 and the 9-pinF to the serial port on the NSX.)

Table 3.1: Cable and Ethernet Transceiver Type for Network Connections

Interface	Connector
FDDI/single-mode FDDI	MIC/ST
IEEE 802.3/Ethernet	15-pin Ethernet AUI (10Base5)
MONITOR/MAINTENANCE ports	25-pin D-type, RS-232
Synchronous serial interface card	25-pin V.35, RS-422
Optical bypass Switch	6-pin DIN

Passport Configuration

The Passport supports routing and bridging mode. For detailed information on configuring the Passport for bridging mode, refer to *Appendix B*.

NetSentry filters must be installed on the Passport. To install these filters on the Passport, you must first copy them from the NSX via FTP. No startup file is required and DPF is not supported.

NOTE

Initial installation of the filter files as well as general Passport installation MUST be accomplished by trained NSG personnel.

The following filters need to be installed under NetSentry on the Passport:

- shun.fil
- first.fil
- last.fil
- incom1.fil
- out1.fil
- other port-specific incom/out filter files

The appropriate filters must be installed on the following filter points:

- first_filter = first
- last_filter = last
- incom1_filter = port 1

NOTE

Numbering ports is the responsibility of the person performing the installation.

- out1_filter = port 1
- other port-specific incomX_filter/outX_filters as needed

A user account with the name netrangr must be installed on the Passport and should have the following characteristics:

UserID:	NETRANGR
Scope:	network
Impact:	debug
nmifs:	local, telnet, fmip, ftp
dir:	/

NOTE

The customer ID for user netrangr must be identical to that of the user account that was initially created/installed on the virtual router on the Passport.

If IP address authentication is enabled on the Passport, the IP address of the NSX must be in the list of authorized addresses.

Install and Configure the NetRanger Director

NetRanger Director Software and Hardware Requirements

Before you begin the NetRanger Director installation process, verify that you meet the following software and hardware requirements.

Software Requirements

The following software must be installed on your workstation:

HP-UX Systems

- HP-UX 10.10 or greater
- HP OpenView 4.1 or greater

Sun Solaris Systems

- Solaris 2.5 or greater
- HP OpenView 4.1 or greater

AIX Systems

- AIX 4.1 or greater
- NetView for AIX 4.1 or greater

Hardware Requirements

Disk Space

The amount of disk space the Director software needs is dictated by the amount of space needed for your network management platform (OpenView or NetView), the amount of space needed for NetRanger logging and database staging, and the amount of space needed for the NetRanger executables and configuration files.

In general, OpenView requires approximately 65 MB in /opt for HP-UX systems and 110 MB in /opt for Solaris systems. NetView for AIX requires approximately 120 MB in /usr. Consult your network management platform documentation for more information about disk space requirements.

NetRanger logging and database staging requires anywhere from 250 MB to 1 GB in /usr/nr/var, depending on the amount of network traffic, type of logging, etc.

NetRanger executables and configuration files will not require more than 30 MB in /usr/nr.

RAM

The RAM requirements for the Director software are dictated by the requirements of the network management software. It is recommended that you run the NetRanger Director on a dedicated machine with at least 64 MB of RAM. Consult your network management platform documentation for more information about RAM requirements and recommendations.

NOTE

If you are installing the NetRanger Director on a workstation that already meets these software and hardware requirements, then you can skip to the *Director Installation* section. Otherwise, you should proceed to the *Pre-Installation* section that corresponds to your operating system.

Pre-Installation

The following sections provide details on pre-installation for HP-UX, Solaris, and AIX systems.

HP-UX Systems and Sun Solaris Systems

Installing HP-UX 10.10 or greater and Installing Solaris 2.5 or greater

- Follow the directions in your HP-UX documentation to either install or upgrade to HP-UX 10.10 or greater.
- Follow the directions in your Sun Solaris documentation to either install or upgrade to Solaris 2.5 or greater.

Installing HP OpenView 4.1 or greater on HP-UX systems

NOTE

HP OpenView will not install correctly if TCP/IP is not functioning properly.

1. **The following parameters must be set before you install HP OpenView:**
   - IP Address
   - Hostname
   - Subnet mask
   - Default gateway hostname
   - Default gateway IP address
   - System time and timezone
2. **To set these parameters, log on as user root, and then run the following command:**

   ```
   /etc/set_parms initial
   ```

3. **Reboot the machine. Once the machine has rebooted, you should be able to**
   - ping your loopback address (**`ping 127.0.0.1`**),
   - ping your IP address (**`ping <IPAddress>`**),
   - resolve your loopback address (**`nslookup 127.0.0.1`**),
   - resolve your IP address (**`nslookup <IPAddress>`**),
   - resolve your hostname (**`nslookup <hostname>`**), and
   - verify that the timezone is correct (**`date`**).

   Do not go to the next step until these TCP/IP parameters are properly configured.

4. **Install HP OpenView 4.1 or greater on the HP-UX system (see the HP OpenView Installation Manual for details).**

5. **Add the following lines to the /.profile for user root. Please note the space between the "." and the "/":**

```
. /opt/OV/bin/ov.envvars.sh
export PATH=$PATH:$OV_BIN
```

NOTE

If user root does not use korn or Bourne shell, then you must translate and place these lines in the appropriate shell configuration file.

Installing HP OpenView 4.1 or greater on Solaris 2.5 or greater

---

NOTE

HP OpenView will not install correctly if TCP/IP is not functioning properly.

---

1. **The following parameters must be set before you install HP OpenView:**
   - IP Address
   - Hostname
   - Subnet mask
   - Default gateway IP Address
   - Default gateway Hostname
   - System time and timezone

2. **After setting the parameters, reboot the machine. Once the machine has rebooted, you should be able to**
   - ping your loopback address (**`ping 127.0.0.1`**),
   - ping your IP address (**`ping <IPAddress>`**),
   - resolve your loopback address (**`nslookup 127.0.0.1`**),
   - resolve your IP address (**`nslookup <IPAddress>`**),
   - resolve your hostname (**`nslookup <hostname>`**), and
   - verify that the timezone is correct (**`date`**).

   Do not go to the next step until these TCP/IP parameters are properly configured.

3. **Install HP OpenView 4.1 or greater on the Sun Solaris machine (see the HP OpenView Installation Manual for details).**

NOTE

The HP OpenView installation will fail if semaphores are not enabled. Please refer to the section entitled Requirements for SunOS and Solaris Systems in the *HP OpenView Network Node Manager Products Installation Guide* to enable semaphores.

NOTE

HP OpenView A.04.10 will not install on Solaris 2.5.x without an OpenView patch. Please contact your authorized HP representative to obtain this patch. (HP OpenView B.04.11 and greater do not require this patch.)

4. **Add the following lines to the /.profile for user root. Please note the space between the "." and the "/":**

```
. /opt/OV/bin/ov.envvars.sh
export PATH=$PATH:$OV_BIN
```

NOTE

If user root does not use korn or Bourne shell, then you must translate and place these lines in the appropriate shell configuration file.

Pre-Installation for IBM AIX Systems

Installing AIX 4.1 or greater

Follow the directions in your IBM AIX documentation to either install or upgrade to AIX 4.1.

Installing NetView for AIX 4.1 or greater on AIX 4.1 or greater

1. **The following parameters must be set before you install NetView for AIX:**
   - IP Address
   - Hostname
   - Subnet mask
   - Default gateway hostname
   - Default gateway IP address
   - System time and timezone

2. **To set these parameters, log on as user root, and then use the smit tool. From the command line, type** `smit tcpip &` **and then select** Minimum Configuration & Startup **from the menu.**
   Change the proper fields and select **OK** to save the new configuration.

3. **Reboot the machine. Once the machine has rebooted, you should be able to**
   - ping your loopback address (**`ping 127.0.0.1`**),
   - ping your IP address (**`ping <IPAddress>`**),
   - resolve your loopback address (**`nslookup 127.0.0.1`**),
   - resolve your IP address (**`nslookup <IPAddress>`**),
   - resolve your hostname (**`nslookup <hostname>`**), and
   - verify that the timezone is correct (**`date`**).

   Do not go to the next step until these TCP/IP parameters are properly configured.

4. **NetView for AIX requires that the En_US locale be installed. To verify that En_US has been installed, enter the following command from root:**

   ```
   locale -a
   ```

   It should show at least: `En_US.IBM-850`

5. **If En_US has not been installed, use the smit tool (`smit mlang &`) and select the** Add Additional Language Environments **menu item. Choose the following in this dialog box:**

   - In the "INPUT device..." select the input device which contains the AIX 4.1 (or greater) media.
   - Select "List" for "CULTURAL..." and select "En_US" (may have IBM-850 with En_US), if present.

NOTE

Ensure you type an upper case E for "En_US".

   - Select "List" for "LANGUAGE..." and select "En_US" (may have IBM-850 with En_US).
   - Select "Do" and it should install the "En_US" locale.

6. **NetView for AIX requires that specific AIX packages be installed (for example: bos.rte, bos.compat.links, X11.compat.fnt.pc). Consult the NetView for AIX installation instructions to ensure that you have all of the required packages installed. To display the list of installed AIX packages on your machine, type the following command:**

   ```
   lslpp -L
   ```

7. **Install NetView for AIX (see the NetView for AIX installation manual for details).**

Director Installation

To install the NetRanger Director software, follow these steps:

1. **If the OpenView user interface is running, stop it now by selecting Map→Exit from the OpenView menu. If other users have other copies of the user interface running and exported to other displays, ask them to shut down the user interface temporarily.**

2. **If you are updating an existing NetRanger system, stop the NetRanger daemons and make a backup of /usr/nr.**

3. **Using `su`, log on as user root.**

4. **Put the NetRanger/Director tape in the tape drive if you have not already done so.**

5. **Go to the /tmp directory by typing**

   ```
   cd /tmp
   ```

6. **Untar the NetRanger/Director files from the tape using the following syntax:**

   ```
   tar -xvf <tape device>
   ```

   Where `<tapedevice>` is the name of the tape drive.

   The NetRanger/Director install tape should contain compressed .tar files whose names have the following format:

   ```
   WGCnsx.<version>.<release>.<modlevel>.<sys type>.tar.Z
   WGCdrctr.<version>.<release>.<modlevel>.<sys type>.tar.Z
   WGCcfgs.<version>.<release>.<modlevel>.<sys type>.tar.Z
   WGCsapd.<version>.<release>.<modlevel>.<sys type>.tar.Z
   JDK_<version>_<release>_<modlevel>-<sys type>.tar.Z
   ```

   The tape will also include the following four utilities:

   ```
   nrInstall
   nrUninstall
   wgcInstall
   wgcUninstall
   ```

7. **Run the NetRanger installation utility by typing**

   ```
   ./nrInstall -D
   ```

8. **Once all of the packages are installed, you will see the message `Installation complete`. At this point, check the /tmp/nrinstall.log file for**

errors.

9. **The Director installation process creates an account for the user netrangr. You must set a password for that user. To set the password, type**

```
passwd netrangr
```

The installation is now complete.

Post-Installation

Configuring the NetRanger Background Processes

The NetRanger background process configuration files enable the Director to communicate with NSX machines on your network. Run the nrconfig utility to configure these files. (Please refer to the *NetRanger Configuration Instructions and Worksheets* section in this chapter for information about the nrconfig utility.)

1. **Stop the NetRanger/Director daemons by typing (as user netrangr)**

   ```
   /usr/nr/bin/nrstop
   ```

2. **Refer to the *NetRanger Configuration Instructions and Worksheets* and run (as user netrangr)**

   ```
   /usr/nr/bin/nrconfig
   ```

3. **After running nrconfig, verify the following in the files in /usr/nr/etc on the Director machine:**

   - Verify that the /usr/nr/etc/auths file lists the Director machine.
   - Verify that the /usr/nr/etc/daemons file has listings for nr.smid, nr.postofficed, nr.configd, and nr.loggerd. If you are using sapd to propagate a relational database, then ensure there is a listing for nr.sapd. If you configured eventd, then ensure there is a listing for nr.eventd. If there is an entry for nr.sensord in this file, ensure that it is commented out. (Use the # key to comment out a line.)
   - Verify that the /usr/nr/etc/hosts file contains an entry for each NSX sensor and routing post office. The localhost entry in the Director's file should match the Director's hostId/orgId pair.
   - Verify that the /usr/nr/etc/organizations file matches the /usr/nr/etc/organizations file on the NSX machines.
   - Verify that the /usr/nr/etc/routes file contains all of the routes files on the NSX machines.
   - Verify that the /usr/nr/etc/services and /usr/nr/etc/signatures files exist.
   - Verify that the /usr/nr/etc/smid.conf file lists *loggerd* as a dupDestination. (This step is optional.)

4. **Verify the following information in the files in /usr/nr/etc for each NSX machine (this will also need to be done for any subsequent NSX machine installations):**

   - Verify that the /usr/nr/etc/destinations file is configured to send alarms to the *smid* process on the Director. ***This is very important. The Director will not receive alarms if this is not done.***
   - Verify that the proper name and address of the Director is in the /usr/nr/etc/routes file.

Configuring the Network Management Background Processes

There are many daemons that are shipped with OpenView/NetView that are not needed for the Director to work.

You can disable these daemons so they do not start when you type `ovstart`. Disabling these daemons provides better performance and response time and makes managing and using OpenView/NetView easier.

NOTE

If you are using OpenView/NetView for IP network management as well as for the Director, then you should **not** disable any daemons.

Disabling Daemons in HP-UX and Sun Solaris

To disable the daemons, follow these steps:

1. **Bring down all copies of the user interface with the** Map→Exit **menu option.**
2. **Log on as user root.**
3. **Stop the OpenView daemons by typing** `ovstop`
4. **Type each of the following commands:**

```
ovdelobj /etc/opt/OV/share/lrf/netmon.lrf
ovdelobj /etc/opt/OV/share/lrf/ovtopmd.lrf
ovdelobj /etc/opt/OV/share/lrf/snmpCollect.lrf
ovdelobj /etc/opt/OV/share/lrf/ovrepld.lrf
ovdelobj /etc/opt/OV/share/lrf/ovactiond.lrf
```

If you need to re-enable the daemons, follow the above steps substituting `ovaddobj` for `ovdelobj`.

Disabling Daemons in AIX

To disable the daemons, follow these steps:

1. **Bring down all copies of the user interface with the** File→Exit **menu option.**
2. **Log on as user root.**
3. **Stop the NetView daemons by typing** `ovstop`.
4. **Type each of the following commands:**

```
ovdelobj /etc/opt/OV/share/lrf/netmon.lrf
ovdelobj /etc/opt/OV/share/lrf/ovtopmd.lrf
ovdelobj /etc/opt/OV/share/lrf/snmpCollect.lrf
ovdelobj /etc/opt/OV/share/lrf/ovrepld.lrf
ovdelobj /etc/opt/OV/share/lrf/ovactiond.lrf
```

   If you need to re-enable the daemons, follow the above steps substituting `ovaddobj` for `ovdelobj`.

Configuring the User Interface for HP-UX and Sun Solaris Systems

The following steps only need to be done once:

1. **Log on as user root.**
2. **Start the OpenView daemons by typing** `ovstart`.
3. **Log on as user netrangr.**
4. **Start the NetRanger daemons by typing** `nrstart`.
5. **Start the user interface by typing** `$OV_BIN/ovw &`.
6. **Double-click the NetRanger icon.**
7. **Choose the menu option** Map→Maps→Describe/Modify.
8. **Under the Compound Status heading, press the** Propagate Most Critical **button.**

9. **Choose** OK.

10. **Choose the menu option** Map→Submaps→Set This Submap As Home.

11. **Choose the menu option** Map→Submaps→Describe/Modify.

12. **Under the Background Graphics: heading, press the** Browse... **button.**

13. **From the pop-up list, select the background graphic of your choice. (The usastates.gif is a popular choice.) You could also create a custom .gif file with any graphics program and use that .gif file as an OpenView submap background.**

14. **Choose** OK, **and then choose** OK **again.**

The following steps should be done every time a new NetRanger/NSX machine is added to your system:

1. **Run nrstop.**

2. **Bring down all copies of the user interface with the** Map→Exit **menu item.**

3. **Run nrconfig to update the configuration files.**

4. **Run nrstart.**

5. **Run** ovw &.

6. **Add the NSX icon to the OpenView map using the following steps:**
   - **Select the** Edit→Add Object **menu item.**
   - **Click on the Net Device icon. Several icons will appear in the bottom of the window.**
   - **Drag the NSX 2000 icon to the NetRanger submap (the submap containing the Director icon) by pressing and holding the middle mouse button while positioning the mouse pointer over the NSX 2000 icon. An Add Object window should appear.**

7. **Choose NetRanger/Director from the list, and press the** Set Object Attributes **button.**

8. **In the hostname field, enter the name of the NSX machine exactly as you entered it in the *usr/nr/etc/hosts* file using the format <hostname>.<organization name>**

9. **Press the** Verify **button. If you entered the hostname correctly, the Organization and Host IDs should have been filled in for you.**

10. Once the hostname, Organization ID, and Host ID are correct, choose OK.

11. Press the Set Selection Name button. Choose the selection name from the list and then press OK in the Selection Name window. Press the OK button in the Add Object Window.

12. You should see the NSX icon turn green. If you double-click on the NSX icon, you should see icons that represent the processes running on that machine.

Configuring the User Interface for AIX Systems

The following steps only need to be done once:

1. **Log on as user root.**
2. **Start the NetView daemons using** `ovstart`.
3. **Log on as user netrangr.**
4. **Start the NetRanger daemons using** `nrstart`.
5. **Start the user interface by typing** `/usr/OV/bin/ovw &`.
6. **Double-click the NetRanger icon.**
7. **Choose the** File→Describe Map **menu item.**
8. **Under the Compound Status heading, press the** Propagate Most Critical **button.**
9. **Choose** OK.
10. **Choose the** Options→Set Home Submap **menu item.**

The following steps should be done every time a new NetRanger/NSX machine is added to your system:

1. **Run nrconfig to update the configuration files.**
2. **Add the NSX icon to the OpenView map using the following steps:**
   - **Select the** Edit→Add→Object **menu item.**
   - **Click on the Net Device icon. Several icons will appear in the bottom of the window.**
   - **Drag the NSX 2000 icon to the NetRanger submap (the submap containing the Director icon) by pressing and holding the middle mouse button while positioning the mouse pointer over the NSX 2000 icon. An** Add Object **window should appear.**
3. **Choose NetRanger/Director from the list, and press the** Set Object Attributes **button.**
4. **In the hostname field, enter the name of the NSX machine exactly as you entered it in the *usr/nr/etc/hosts* file.**

5. **Press the** Verify **button. If you entered the hostname correctly, the Organization and Host IDs should have been filled in for you.**

6. **Once the hostname, Organization ID, and Host ID are correct, choose** OK.

7. **Press the** Set Selection Name **button. Choose the selection name from the list and then press** OK **in the Selection Name window. Press the** OK **button in the** Add Object **window.**

8. **You should see the NSX icon turn green. If you double-click on the NSX icon, you should see icons that represent the processes running on that machine.**

Configuring New Users

By default, user netrangr is the only user configured to use and reconfigure the NetRanger Director system. If you want to grant Director software access to another user, you must add the user to the UNIX group netrangr. You must also configure the user's shell environment appropriately. Instructions for both are listed below.

Adding Users to the netrangr Group

HP-UX—On HP systems, users can be added to and removed from groups using the sam utility.

NOTE

On HP Systems, if a user is in the group netrangr (but netrangr is not that user's primary group), then the user must type `newgrp - netrangr` to execute nrdlmap.

Sun Solaris—On Sun systems, use the admintool to add users to and remove users from groups.

AIX—On AIX systems, users can be added to and removed from groups using the smit utility.

Configuring the User Environment

User netrangr uses the ksh UNIX shell. The environment settings for user netrangr are kept in the file /usr/nr/.profile. The .profile puts /usr/nr/bin in the $PATH, and then it sets environment variables for OpenView, Java, and Oracle.

Complete the NetRanger Configuration Instructions and Worksheets

The NetRanger configuration utility (**nrconfig**) generates the NetRanger NSX, Director, and BorderGuard configuration files. Use the following worksheets as a guide for gathering information for nrconfig (such as IP addresses, passwords, and names of network components).

You may find it convenient to make copies of these worksheets when you are ready to configure your NetRanger software.

NetRanger Configuration Instructions and Worksheets

Version 1.3.1

The Director and NSX systems are configured with a utility called nrconfig. In addition to running this utility at installation, you can run this utility at any time to change an existing configuration. This section includes worksheets to help you gather the information (such as IP addresses, passwords, and names of network components) you need before you run nrconfig.

NOTE

nrconfig does not retain changes manually made to the files in the /usr/nr/etc or /temp directories because presently it is an installation tool and not intended to be an ongoing configuration interface for NetRanger Configuration Files.

Before you run nrconfig, you must have completed the installation for one or more of the following NetRanger components:

- **The NSX (WGCnsx)**
- **The Director (WGCnsx, WGCdrctr, WGCcfgs)**
- **The optional Database/File Management (WGCsapd) software**

New NSX systems are shipped with the packages installed, so you only need to install the packages on Director systems or NSXs that you are upgrading.

You must also have gathered the following information about your network:

- **IP addresses of all network components**
- **Names of all network components**
- **Services you wish to allow in and out of your network**
- **NetRanger Organization IDs and Names**
- **Routing Information (both IP and NetRanger)**
- **Passwords and other host information**

NOTE

You must run nrconfig as user netrangr. nrconfig can be run against an active NSX or Director system without having to shut down any of the NetRanger daemon services.

The NetRanger Configuration Program

To run nrconfig, type this command:

```
/usr/nr/bin/nrconfig
```

nrconfig initially displays the following prompt:

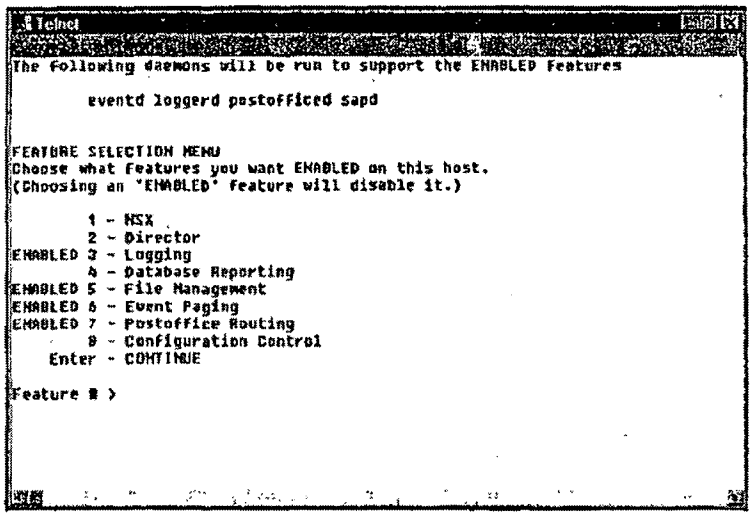
```
Are you ready to continue with configuration of your NetRanger? (y/n)>
```

If you have gathered the required information and are ready to configure the NetRanger software, choose **y** then press **Enter** to continue.

FEATURE SELECTION MENU

This menu, which is shown below, prompts you to select which features you want this installation to support.

When you select a feature from the menu, it is prefixed by "ENABLED" to indicate that it is presently selected. To disable a selected feature, simply choose it from the menu again. The list of daemons required to support the "ENABLED" features is shown at the top of the screen above the Feature Selection menu.



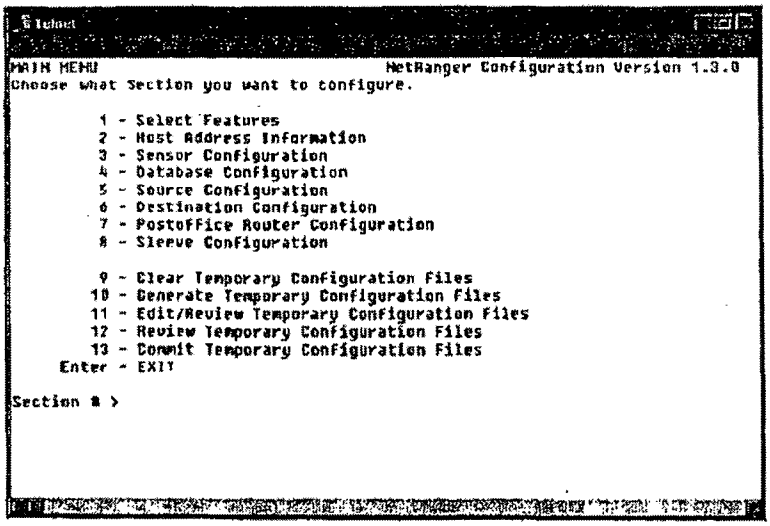
To continue with NetRanger Configuration, choose **Enter** at the menu prompt. This takes you to the Main menu. You can return to this or any other menu at any time from the Main menu.

NOTE

For Standard NSX installation, select features 1 and 5; for standard Director installation, select features 2, 6, and 8.

MAIN MENU

Each menu item takes you through a series of subordinate, or "child", menus that are organized by feature requirements. Features that are not required for the specified NetRanger configuration are preceded by N/A (Not Available). You can return to a configuration menu item at any time by selecting option 1 from the Main menu without losing any information that you may have already input.



The following pages contain worksheets to help you organize your nrconfig information. Each worksheet identifies the required features. You can skip those pages that discuss features that have not been "ENABLED".

1 - Select Features

This menu item takes you to the Feature Selection menu where you can chose which features you want to enable on the NetRanger you are configuring.

2 - Host Address Information **(Required for all installations)**

LOCAL HOST ADDRESS MENU

Use this menu to enter the Organization names and IDs and Host names and IDs for the NetRanger you are configuring.

Field Name	Description	Your Entry
Organization Name	This is the symbolic name for the Organization ID. The organization name must be identical on all NetRanger systems (NSXs and Directors) within an organization.	
Organization ID	This is the organization ID for the NetRanger you are configuring. You must enter a value between 1-65535 in this field. This ID must be identical on all NetRanger systems (NSXs and Directors) within an organization. To obtain a globally unique organization ID, call WheelGroup Corporation at 888-WGC-NSOC.	
Host Name	A symbolic name for the NetRanger component you are configuring. *This is NOT the network host name.*	
Host ID	A unique ID for the NetRanger you are configuring. *This is NOT the IP address.* You must enter a value between 1-65535 in this field.	

3 - Sensor Configuration **(Required for NSX)**

BORDERGUARD TYPE MENU

Use this menu to enter BorderGuard configuration data in the fields.

Field Name	Description	Your Entry
BorderGuard's Network Host Name	This is the network host name for the BorderGuard used in /etc/hosts on the NSX or in DNS.	
BorderGuard's PASSWORD	This is the password used to log into the BorderGuard.	**DO NOT WRITE YOUR PASSWORD HERE!**
BorderGuard's Version ID/Mode	This is the BorderGuard's Version ID and Configuration Mode. The BorderGuard Version 3 software only supports Router mode. The BorderGuard Version 4 NetSentry software supports either Router or Bridge mode. The default is Version 4 Bridge Mode.	

Based on the **BorderGuard's Version ID/Mode**, the following configuration menus refer either to *Router Mode* or *Bridge Mode Configuration.*

Router Mode Configuration

This section establishes the IP addresses and netmasks for each of the BorderGuard's network interfaces (the BorderGuard should separate your internal protected networks from outside untrusted networks).

The LAN Interfaces Entry menu is the first example of a configuration menu that allows you to list multiple entries. You can add as many entries to the list as you can see on your screen. Each Entry menu allows you to add, edit, or delete entries in the list.

LAN INTERFACES ENTRY MENU

Use this menu to enter BorderGuard configuration information for a LAN interface.

NOTE
It is important to list the outside untrusted network interface first!

Field Name	Description	Your Entry(ies)
Interface's IP Address	This is the IP address used by the BorderGuard on the subnet connected to this interface.	
Netmask	This is the IP mask used on the subnet connected to this interface.	

WAN INTERFACES ENTRY MENU

Use this menu to enter BorderGuard configuration information for a WAN interface.

Field Name	Description	Your Entry(ies)
Interface's IP Address	This is the IP address used by the BorderGuard for this PPP interface.	
PPP IP Address	This is the Destination IP address used by the BorderGuard for this PPP interface.	

BORDERGUARD CONFIGURATION MENU

Use this menu to enter configuration data for a BorderGuard that will be acting as a router.

Field Name	Description	Your Entry
BorderGuard's Primary IP Address*	This is the IP address the NetRanger uses to establish encrypted sleeves for communicating to other NetRanger post offices. This defines the IP address at this end of the encrypted sleeve. Generally this should be the BorderGuard's IP address on the network it is protecting.	
BorderGuard's default gateway	This is the IP address that the BorderGuard should use for its default gateway for IP packet routing.	
Minutes to log on an event	This is the number of minutes to log IP traffic after a serious event occurs. The recommended value is 15 minutes.	
Minutes to shun on an event	This is the length of time in minutes that traffic should be blocked when a serious event occurs. The recommended value is 1440, which equals one day.	
NSX IP Address	This is the IP address used by the NSX on the subnet connecting the NSX to the BorderGuard.	
BorderGuard's IP Address connected to NSX	This is the IP address used by the BorderGuard on the subnet connecting the NSX to the BorderGuard.	
BorderGuard's External IP Address	The NetRanger uses the External IP address as the connection to the untrusted networks.	

NOTE
*If you are using encrypted sleeves over the Internet, this should be a routeable Internet address.

STATIC ROUTES ENTRY MENU

Use this menu to enter the IP addresses, Netmasks, and Gateway IP addresses for the static routes to be implemented by the BorderGuard.

Field Name	Description	Your Entry(ies)
Network's IP Address	This is the IP address for the subnet for the static route.	
Network's Netmask	This is the netmask for the subnet.	
Network's Gateway IP Address	This is the IP address that acts as a gateway to the subnet for the static route.	

Bridge Mode Configuration

BORDERGUARD CONFIGURATION MENU

Use this menu to enter configuration data for a BorderGuard that will be acting as a bridge.

Field Name	Description	Your Entry
BorderGuard's IP Address*	The NetRanger system uses this IP address to establish encrypted sleeves and to communicate with the NSX. This is the BorderGuard's IP address on the subnet connecting the NSX to the BorderGuard.	
BorderGuard's default gateway	This is the IP address that the BorderGuard should use for its default gateway for IP packet routing.	
Minutes to log on an event	This is the number of minutes to log IP traffic after a serious event occurs. The recommended value is 15 minutes.	
Minutes to shun on an event	This is the length of time in minutes that traffic should be blocked when a serious event occurs. The recommended value is 1440, which equals one day.	
NSX IP Address	This is the IP address used by the NSX on the subnet connecting the NSX to the BorderGuard.	

NOTE

*If you are using encrypted sleeves over the Internet, this should be a routeable Internet address.

The following two sensor configuration menus refer to both *Router Mode* and *Bridge Mode* configuration.

SECURITY POLICY CONFIGURATION MENU

Use this menu to establish which incoming services to allow on your interface. You also use this menu to specify the servers to which this traffic will be allowed to pass.

Field Name	Description	Your Entry(ies)
Gateway's IP Address*	This is the IP address of the server that is allowed to service requests coming in through the BorderGuard's External IP address.	
Port	This is the port on the server for the allowed service.	

NOTE

This field supports the definition of multiple IP addresses using any combination of the following formats:

```
ipAddr - a single IP address.
ipAddr,ipAddr - multiple IP addresses.
ipAddr..ipAddr - an inclusive range of IP addresses.
```

Examples: 10.1.6.1, 10.1.6.20, 10.1.6.31, and 10.1.6.35 define 10.1.6.10, 10.1.6.20, and the IP addresses from 10.1.6.31 through 1.01.6.35.

INTERNAL NETWORKS ENTRY MENU

Use this menu to establish the IP addresses and Netmasks for the Internal Protected Networks.

Field Name	Description	Your Entry(ies)
Network's IP Address	This is the IP address for the subnet for the internal network.	
Network's Netmask	This is the netmask for the subnet.	

4 - Database Configuration **(Required for Database)**

DATABASE CONFIGURATION MENU

Use this menu to enter the Database User ID, the Database Password, and the person to be notified for NetRanger events.

Field Name	Description	Your Entry
Database USER ID	This is the user ID used to log into the database.	
Database PASSWORD	This is the password used to log into the database.	**DO NOT WRITE YOUR PASSWORD HERE!**
Notify Person*	This is the person the NetRanger system notifies. Notification is based on criteria you will configure in the sapd.conf file during installation and configuration of NetRanger's *sapd* component. Please refer to *Chapter 5* in this *User's Guide* for additional information.	

NOTE
*This entry must be a valid e-mail or pager address.

5 - Source Configuration **(Required for Director)**

SOURCE ENTRY MENU

Use this menu to enter the Organization and Host names, Organization and Host IDs, and IP routing addresses for the sources of NetRanger events. Enter this information for each NSX that will be sending events to the Director.

Field Name	Description	Your Entry(ies)
Organization Name	This is the source's organization name.	
Organization ID	This is the source's organization ID. This ID must be identical on all NetRanger systems (NSXs and Directors) within an organization. You must enter a value between 1-65535 in this field.	
Host Name	This is the source's host name. This is NOT the network Host name.	
Host ID	This is the source's host ID. You must enter a value between 1-65535 in this field. This is NOT the IP address.	
IP Address to route through	This is the IP address of the NetRanger postoffice that can route NetRanger packets from the source. (If the network does not require an intermediary postoffice, this should be the IP address of the source.)	

6 - Destination Configuration **(Required for NSX/Optional for the Director)**

DESTINATION ENTRY MENU

Use this menu to enter the Organization and Host names, Organization and Host IDs, IP routing addresses, Destination Services, and Event Logging Levels for the destinations of NetRanger events.

Field Name	Description	Your Entry(ies)
Organization Name	This is the destination's organization name.	
Organization ID	This is the destination's organization ID. This ID must be identical on all NetRanger systems (NSXs and Directors) within an organization. You must enter a value between 1-65535 in this field.	
Host Name	This is the destination's host name. *This is NOT the network host name.*	
Host ID	This is the destination's host ID. You must enter a value between 1-65535 in this field. *This is NOT the IP address.*	
IP Address to route through	This is the IP address of the NetRanger postoffice that can route NetRanger packets to the destination. (If the network does not require an intermediary postoffice, this should be the IP address of the destination.)	
Service	This is the name of the service NetRanger events will be sent to. You must enter *loggerd*, *smid*, or *eventd* in this field.	
Level	This is the lowest level of NetRanger alarm/event to send to the service (*loggerd*, *smid*, or *eventd*) you chose in the previous field. You must enter a value between 1-255 in this field. (The recommended level for alarms sent to *smid* is 2 and 1 for alarms sent to *loggerd*.)	

NOTE

On an NSX, you should add the Director machine as a *smid* destination and you should add the NSX machine as a *loggerd* destination so that you can log level 1 alarms/events on the NSX.

7 - Postoffice Router Configuration **(Required for Postoffice routing)**

ROUTER ENTRY MENU

Use this menu to enter the Organization and Host names, Organization and Host IDs, and IP Routing Addresses for remote NetRanger nodes that are not being used as a source or destination.

Field Name	Description	Your Entry
Organization Name	This is the organization name of the remote NetRanger node.	
Organization ID	This is the remote NetRanger node's organization ID. This ID must be identical on all NetRanger systems (NSXs and Directors) within an organization. You must enter a value between 1-65535 in this field.	
Host Name	This is the remote NetRanger node's host name. *This is NOT the network host name.*	
Host ID	This is the remote NetRanger node's host ID. You must enter a value between 1-65535 in this field. *This is NOT the IP address.*	
IP Address to route through	This is the IP address of the NetRanger postoffice that can route NetRanger packets to the remote node. (If the network does not require an intermediary postoffice, this should be the IP address of the remote node.)	

8 - Sleeve Configuration **(Optional for NSX)**

SLEEVED NETWORK ENTRY MENU

Use this menu to enter the Remote Organization ID, Remote IP routing addresses, and Remote Network Netmasks for Sleeved Networks.

Field Name	Description	Your Entry(ies)
Sleeve Remote Org ID	This is the organization ID for the remote end of the sleeve.	
Sleeve Remote IP Address	This is the IP address for the remote end of the sleeve.	
Sleeve Remote Netmask	This is the subnet netmask for the remote end of the sleeve.	

9 - Clear Temporary Configuration Files

This menu item prompts you to ensure that you want to clear the temporary configuration files for the NetRanger software.

```
Are you sure you want to CLEAR the Temporary Configuration files? (y/n)>
```

Choose **y** to clear and reinitialize the temporary NetRanger configuration files in /usr/nr/etc/wgc and the temporary BorderGuard configuration files in /usr/nr/etc/nsc to their default values. This will not discard any configuration information you entered in the current nrconfig session.

10 - Generate Temporary Configuration Files

This menu item prompts you to insure that you want to generate the temporary configuration files for the NetRanger software.

```
Are you sure you want to GENERATE the Temporary Configuration files? (y/n)>
```

Choose **y** to write the temporary NetRanger configuration files containing all the modifications made in the current nrconfig session to /usr/nr/etc/wgc and the temporary BorderGuard configuration files to /usr/nr/etc/nsc.

NOTE

You should review the temporary NetRanger configuration files located in the /usr/nr/etc/wgc directory and the BorderGuard configuration files located in the /usr/nr/etc/nsc directory after nrconfig has generated the temporary configuration data. The temporary NetRanger configuration files **must** be committed to /usr/nr/etc and the temporary BorderGuard configuration files to /tmp after review or after any manual changes. The BorderGuard files must then be loaded onto the NSG BorderGuard via a TFTP session initiated by the BorderGuard.

11 - Edit/Review Temporary Configuration Files

This menu item starts a vi edit on the temporary NetRanger configuration files in /usr/nr/etc/wgc and the temporary BorderGuard configuration files to /usr/nr/etc/nsc.

12 - Review Temporary Configuration Files

This menu item starts a "more" command on the temporary NetRanger configuration files in /usr/nr/etc/wgc and the temporary BorderGuard configuration files in /usr/nr/etc/nsc.

13 - Commit Temporary Configuration Files

This menu item prompts you to ensure that you want to commit the temporary configuration files for the NetRanger software to the NetRanger Configuration File Directory.

```
Are you sure you want to COMMIT the Temporary Configuration files to the
NetRanger Configuration File Directory '/usr/nr/etc' and to the BorderGuard
Configuration File Directory '/tmp'? (y/n)>
```

Choose **y** to write the configuration temporary NetRanger configuration files to the /usr/nr/etc and /tmp directories.

WARNING

This overwrites working NetRanger configuration files.

Enter - EXIT

This menu item prompts you to ensure that you are ready to exit nrconfig. To exit, press **Enter**.

```
Are you sure you want to EXIT? (y/n)>
```

Choose `y` to exit nrconfig.

NOTE
You can exit and restart nrconfig without losing any of the configuration information you have input.

NetRanger network configuration is complete.

NOTE
The NetRanger processes **must** be set using /usr/nr/bin/nrhup before the committed NetRanger configuration files will take effect.

NSX Installations
The BorderGuard Configuration Files **must** be uploaded to the BorderGuard and the BorderGuard **must** be restarted before any changes to the configuration files take effect. If you are using a BorderGuard 2000 with Version 4.0 of the NetSentry software, then you may not have enough file space on the BorderGuard boot diskette to load the new BorderGuard configuration files. After you make backup copies of the files on the diskette, then you may delete the *readme*, *firewall.def*, and *firewall.set* files which are not needed for normal operation.

Define the Security Policy

Your security policy defines what type of activities and services you want to allow and disallow at key access points on your network. This includes

- services to **allow in** from untrusted sites,
- services to **allow out** from trusted sites (i.e., internal users), and
- services you want to track, such as Web traffic or FTP usage.

Implementation of your security policy is based largely on the filters installed on your BorderGuard or Passport security device. Use the following sections on **ICMP**, **TCP**, and **UDP** protocols as a guide to the configuration of the filter templates. For more information on these services, you may want to refer to such classic texts as *Internetworking With TCP/IP—Volume 1* by Douglas E. Comer (1995).

After defining your security policy, it may be necessary to manually edit the filter files nrconfig places in /tmp, transfer them to the BorderGuard, and then restart the BorderGuard. By default, the incom1.fil filter is applied to the BorderGuard interface on the untrusted network. If you determine that your security policy is identical to the recommended policy outlined in the worksheets on the following pages, then no changes need to be made to the incom1.fil filter file.

WARNING

All manual edits to the files in /usr/nr/etc (the NetRanger configuration files) and /tmp (the BorderGuard configuration files) will be lost if nrconfig is run and the temporary configuration files are committed.

The following information should be available for each configured NetRanger NSX system:

ICMP (Internet Control Message Protocol)

ICMP allows routers and hosts to send error and control messages to other routers or hosts. One of the most frequent uses of this service is in support of the "ping" command, which queries a remote host or network device to see if it is alive on the network. Hackers commonly use this service to discover potential targets by mapping out a remote network. It also allows internal users to check connectivity to a remote site. Use the following chart to help you map allowable ICMP messages.

Enter "A" for Allowed or "B" for Blocked. All incoming requests to your network should be blocked.

Message	Your Entry	Recommendation	Type Field Offset
Echo Reply		Allow	0
Destination Unreachable		Allow	3
Source Quench		Allow	4
Redirect (change a route)		Allow	5
Echo Request		Block	8
Time Exceeded for a Datagram		Allow	11
Parameter Problem on a Datagram		Allow	12
Timestamp Request		Block	13
Timestamp Reply		Allow	14
Address Mask Request		Block	17
Address Mask Reply		Allow	18

NOTE

You may want to unblock services destined for your Internet servers such as a Web server, mail server, or FTP server. This function is explained in greater detail in *Appendix B.*

To adjust the ICMP security policy, edit the INCOM1_ICMP_FAIL filter in tmp/incom1.fil, transfer the edited file to the BorderGuard, and then restart the BorderGuard. (Refer to the *BorderGuard Configuration* section in this chapter for detailed information about transferring files to the BorderGuard from the NSX.)

TCP (Transmission Control Protocol)

TCP is the most common transport layer protocol used on Ethernet and the Internet. Because of the connection-oriented characteristics of TCP, a connection is established every time a TCP service is used. Therefore, it is easy to ***block certain services from entering*** your network while at the same time ***allowing outgoing*** traffic. Use the following chart to help you map allowable TCP services. This list is not all-inclusive, but rather presents the most common TCP services that are included in the filter templates. Services that can be dynamically added to a filter have a corresponding entry in the NSX's *sensord.conf* file.

Enter "A" for Allowed or "B" for Blocked. Also enter the IP Address of the host you want to allow the service to.

Service	Port	Your Entry	Recommendation	IP Address	Filter Name
FTP Reply	Source Port 20*		Allow (All IPs allowed FTP*)		INCOM1_TCP
FTP	21		Allow 1 IP†		INCOM1_21_TCP_FAIL
Telnet	23		Block		INCOM1_23_TCP_FAIL
SMTP (Mail)	25		Allow		INCOM1_25_TCP_FAIL
DNS	53		Allow		INCOM1_53_TCP_FAIL
Gopher	70		Block		INCOM1_70_TCP_FAIL
Finger	79		Block		INCOM1_79_TCP_FAIL
WWW	80		Allow		INCOM1_80_TCP_FAIL
POP2 (Mail)	109		Block		INCOM1_109_TCP_FAIL
POP3 (Mail)	110		Block		INCOM1_110_TCP_FAIL
RPC	111		Block		INCOM1_111_TCP_FAIL
Auth	113†		Allow 1 or more IP‡		INCOM1_113_TCP_FAIL
NNTP (News)	119		Block		INCOM1_119_TCP_FAIL
NTP	123		Block		INCOM1_123_TCP_FAIL
Exec	512		Block		INCOM1_512_TCP_FAIL
Login	513		Block		INCOM1_513_TCP_FAIL
Cmd	514		Block		INCOM1_514_TCP_FAIL
Printer	515		Block		INCOM1_515_TCP_FAIL

Service	Port	Your Entry	Recommendation	IP Address	Filter Name
ntalk	518		Block		INCOM1_518_TCP_FAIL
uucp	540		Block		INCOM1_540_TCP_FAIL
X11	6000–6063‡		Block		INCOM1_X11_FAIL

*TCP Source Port 20 must be allowed to all hosts that need to be able to FTP to the Internet. Most of the time, this includes everyone, so this TCP service is allowed through.

†Restrict these services to only those host(s) that provide them (i.e., Web server, mail server).

‡The authentication service needs to be allowed for connection back to the mail server and occasionally to individual hosts. Many mail services require this service before they will accept mail, so any trusted system sending mail must allow this service back to it. Some Web sites also require this service. Allowing everyone to use this service does not create any known security holes.

To adjust the TCP security policy, edit the appropriate filter in /tmp/incom1.fil, transfer the edited file to the BorderGuard, and restart the BorderGuard. (Refer to the *BorderGuard Configuration* section in this chapter for detailed information about transferring files to the BorderGuard from the NSX.)

UDP (User Datagram Protocol)

Very few UDP services should be allowed between your network and untrusted sites. UDP is a connectionless service, meaning that it is impossible to distinguish between session initiation and general session data. Use the following chart to help you map allowable UDP services. This list is not all-inclusive, but rather lists the most common UDP services that are included in the filter templates. Other services can be added to the filter and will need to have a corresponding entry in the *sensord.conf* file. The configuration of this file is discussed in a later section.

Enter "A" for Allowed or "B" for Blocked. Also enter the IP Address of the host you want to allow the service to.

Service	Port	Your Entry	Recommendation	IP Address	Filter Name
DNS	53		Allow*		INCOM1_UDP
TFTP	69		Block		INCOM1_TFTP_FAIL
RPC	111		Block		INCOM1_UDP_RPC_FAIL
NTP	123		Block		INCOM1_UDP_NTP_FAIL
NetBIOS	137–139		Block		INCOM1_NETBIOS_FAIL
SNMP	161 & 162		Block		INCOM1_SNMP_FAIL
Syslog	514		Block		INCOM1_SYSLOG_FAIL
RIP	520		Block†		INCOM1_UDP_RIP_FAIL
Response Ports	Port # > 1023		Allow‡		INCOM1_UDP

*You will typically want to restrict external access to a single DNS server.

†If using NetRanger between corporate partners or internal business networks, allow RIP or other routing protocols to pass. Work with the network administrator to determine which protocols need to be allowed.

‡The only standard high-level UDP port usually allowed from an untrusted site is to a DNS server. Restrict it to that single IP address and to the UDP source port 53.

To adjust the UDP security polity, edit the appropriate filter in /tmp/incom1.fil, transfer the edited file to the BorderGuard and restart the BorderGuard. (Refer to the *BorderGuard Configuration* section in this chapter for detailed information about transferring files to the BorderGuard from the NSX.)

OPERATING NETRANGER

4

Working with the Director and the NSX

The NetRanger Director is the Graphical User Interface (GUI) for the NetRanger system. The NetRanger Director (also called "the Director")

- provides a graphical, intuitive display of information pertaining to network security violations in real time;
- displays a hierarchical map of the remote NetRanger software and hardware (e.g., the Sensor processes and the NSX hardware) that send security notifications to the Director;
- provides utilities for configuration of the remote NetRanger applications; and
- provides utilities to query the database of historical security events.

The Director uses popular network management platforms like HP OpenView and IBM NetView to display network security information. As a result of this integration, network management personnel do not have to learn multiple-user interface applications and paradigms to perform different network management tasks.

When a process on a remote NSX machine detects a security violation, a notification (called an "event") is sent from the NSX machine to the Director machine. The Director ensures that the machine and application that generated the event are represented on the graphical map, and then, if the event's severity level exceeds a user-definable threshold, the Director creates an Alarm icon on the map. The color of the Alarm icon is based on the severity of the event. The Application and Machine icons also change color, so it is easy to determine at a glance which machine detected the problem. With a few mouse clicks, details about the Alarm (e.g., source and destination IP address) can be displayed. Location functions can be used to locate Alarms with specific properties.

Once an Alarm is diagnosed and addressed, the user can delete the Alarm icon from the user interface. The Application and Machine icons then revert to their previous state.

Architecture

The NetRanger Director is not a single computer program, but is rather a *set* of applications and background processes that work with a network management platform. The diagram below illustrates the data flow between the processes in NetRanger.

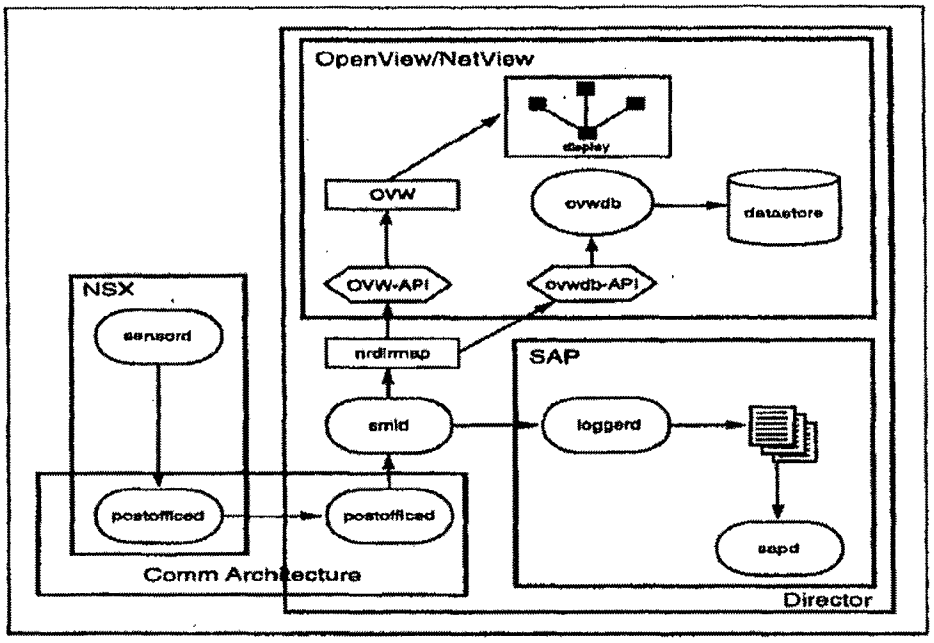


Figure 4.1: The NetRanger Director Architecture

In Figure 4.1, ovals represent background processes, squares represent foreground applications, cylinders represent datastores, hexagons represent APIs, and lines represent the flow of event data. Note that *ovw* and *ovwdb* are part of OpenView/ NetView, *nrdirmap*, *smid*, and *loggerd* are part of the Director, and *sensord* is part of the NSX. Also note that the NSX and the Director both contain *postofficed* processes.

When the *sensord* process detects activity of interest, it generates an event that is sent via the *postofficed* daemons to the *smid* daemon on the Director machine. The *smid* daemon passes the event information to nrdirmap and the *loggerd* daemon, which logs the information.

nrdirmap looks at the severity level of the event. If the event severity exceeds a user-specified level, then *nrdirmap* tells *ovw* to draw an alarm icon. *nrdirmap* also tells *ovwdb* to create an alarm database object in the OpenView/NetView datastore.

Basic Director Functions

Starting the Director

The Director consists of three separate subsystems:

- The NetRanger background processes.
- The network management platform background processes.
- The network management platform user interface.

NOTE

These subsystems should be started in the order listed above to ensure proper operation of the Director.

Starting the NetRanger Background Processes

The NetRanger background processes are configured to start automatically when the machine is rebooted, so in most cases, you will not need to manually start these processes.

In the event that you must start the background processes manually, follow these steps:

1. **Log in as someone in the group** `netrangr`**, and then type**

   ```
   nrstart
   ```

2. **If the executable is not found, then type the fully qualified name,**

   ```
   (/usr/nr/bin/nrstart)
   ```

   put `/usr/nr/bin` **in your path, or type**

   ```
   . /usr/nr/.profile.
   ```

Starting the Network Management Background Processes

Like the NetRanger background processes, the network management platform background processes are configured to start automatically when the machine is rebooted, so in most cases, you will not need to manually start these processes.

In the event that you must start them manually, log in as root and then type `ovstart`

If the executable is not found, then the subdirectory that includes the network management binaries is probably not in your path (the location for OpenView binaries is `$OV_BIN`, and the location for NetView binaries is `/usr/OV/bin`). (To set your path, type `. /usr/nr/.profile`) Consult your network management documentation if you have difficulty starting the network management background processes.

Starting the Network Management User Interface

To start the Director's network management user interface in HP OpenView, log in as a user that belongs to the group `netrangr` and then type

```
ovw &
```

NOTE

The *nrdlrmap* program will start automatically when you bring up the network management user interface. You will never have to manually start *nrdlrmap*. If you use IBM NetView, type `nv6000 &`

Stopping the Director

To stop the Director, stop the subsystems in the *opposite* order in which they were started.

Stopping the Network Management User Interface

If you use HP OpenView, select **Map...Exit** from the menu.

Usually, you will only want to close the user interface. In most circumstances, you will not want to close the background processes.

If you do want to close the background process, log in as user root and then type `ovstop`

If the executable is not found, then the subdirectory that includes the network management binaries is probably not in your path (the location for OpenView binaries is `$OV_BIN`, and the location for NetView binaries is `/usr/OV/bin`). Consult your network management documentation if you have difficulty starting the network management background processes.

Stopping the NetRanger Background Processes

To stop the NetRanger background processes, follow these steps:

1. **Log in as someone in the group** `netrangr` **and then type**

   ```
   nrstop
   ```

2. **If the executable is not found, then either type the fully qualified name**

   ```
   (/usr/nr/bin/nrstop)
   ```

   or put `/usr/nr/bin` in your path.

Checking the Status of the Director Processes

To check the status of all Director processes, follow these steps:

1. **To ensure that the network management background processes are running correctly, type**

   ```
   ovstatus
   ```

2. **To ensure that the NetRanger background processes are running correctly, type**

   ```
   nrstatus
   ```

   If either of these executables cannot be found, check your path.

Understanding the Director's Submap Hierarchy

When you double-click on a symbol, a submap is opened. This submap could have many symbols on it, and you can double-click these symbols to reveal more submaps. These descending submaps can be thought of as an upside-down tree with more and more branches. This upside-down tree structure is called the "submap hierarchy."

Traversing the submap hierarchy that nrdirmap creates is easy once you understand the following structure:

This type of submap...	can contain these symbols:
Root	Collection
Collection	Machines (NSX and Director)
	Collections
	Connections
Machine	Applications
Application	Alarms
	Alarm Sets

NOTE

Collections and Alarms do not have submaps. They represent the "leaves" in the submap tree.

Figure 4.2 illustrates the root submap. It is the highest level submap in the hierarchy. The root submap has no "parent submap". On the root submap, there should be a symbol representing a Collection of machines.

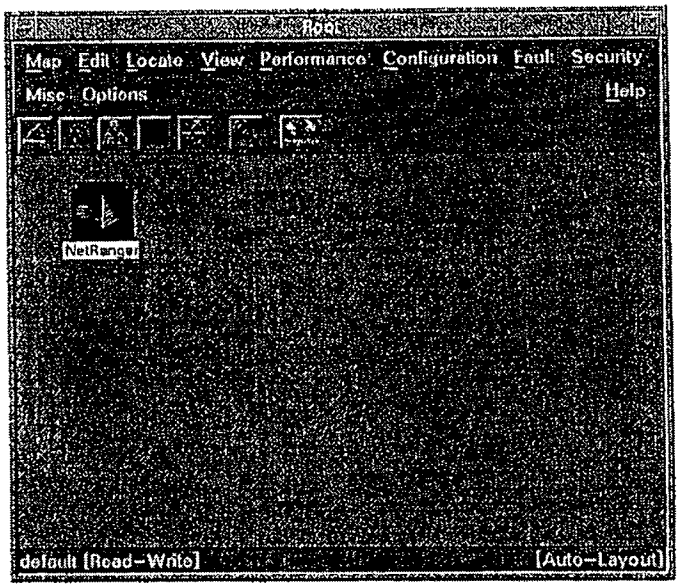


Figure 4.2: The NetRanger Director Submap

When you double-click on a Collection symbol like the one shown in Figure 4.2, a Collection submap is displayed. A Collection submap can have NSX Machines, the Director Machine, other Collection symbols, and Connections between Machines. The Collection submap shown in Figure 4.3 contains a Director machine symbol only

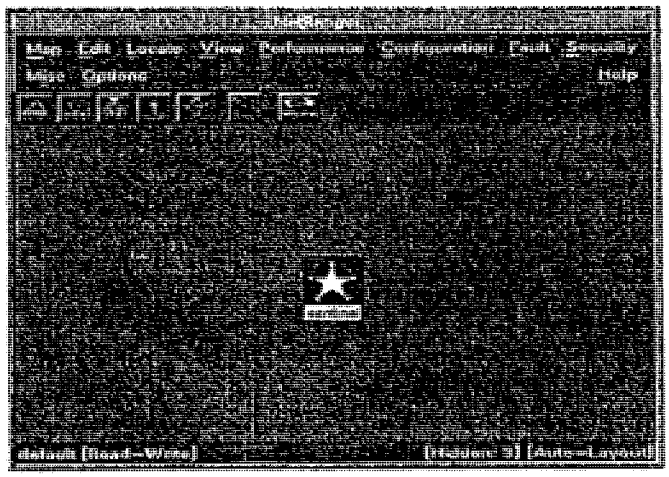


Figure 4.3: A Collection Submap

When you double-click on a Machine symbol, a Machine submap is displayed. A Machine submap contains symbols that represent the different applications running on the machine. Refer to Chapter 1 in this *User's Guide* for an overview of these application daemon services.

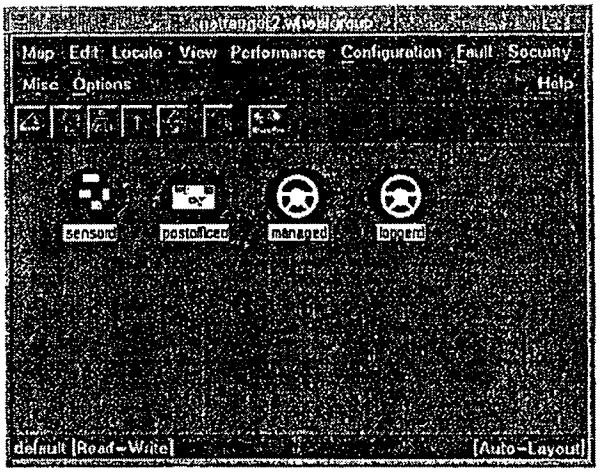


Figure 4.4: A Machine Submap

When you double-click on an Application symbol, an Application submap is displayed. An application submap contains alarms that application had generated. For instance, if the *sensord* application for a machine generates an event and sends it to the Director, the Director will draw an alarm icon on the submap belonging to that machine's application

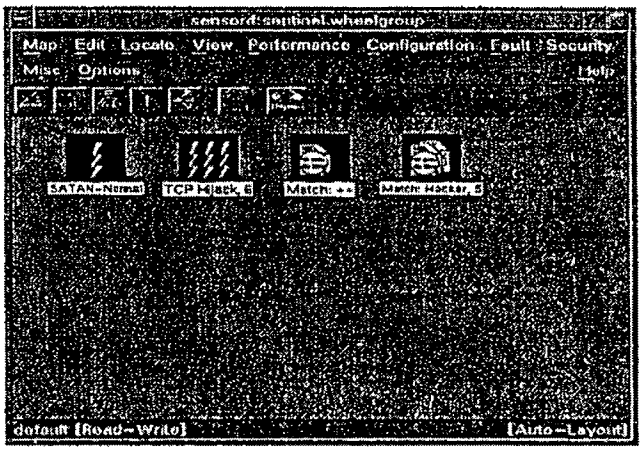


Figure 4.5: An Application Submap

For most alarms, the label under the alarm symbol will match the alarm's "Alarm Name" attribute. For instance, an alarm with the Alarm Name "Net Sweep" will have a "Net Sweep" symbol label.

Alarms with the name "String Match" and "Sec Violation" (Security Violation) will have their symbol labels taken from the "Alarm Details" attribute. This is because there are many types of Security Violations, and there are an infinite number of potential string matches, so for these two alarm types, the Alarm Name itself is not specific enough. For Security Violation alarms, the label will match the name of the specific violation, and for String Matches, the label will be the string that was detected.

Application submaps can also contain a special Alarm symbol called an "Alarm Set". An Alarm Set is created when multiple alarms are received that are identical in all respects *except for* timestamp *and* sequence number. For example, if you get 20 string match alarms with the same attributes (source and destination address, source and destination port etc.), then the 20 alarms will be represented by a single Alarm Set symbol.

Alarm Sets can be differentiated from Alarms in two ways. First, the end of an Alarm Set symbol label will have a comma followed by the number of alarms represented by the Set. Second, the Alarm Set symbol type is slightly different from an Alarm symbol type. An Intrusion Alarm icon has one lightening bolt and an Intrusion Alarm Set has multiple lightening bolts. A String Match Alarm icon has one sheet of paper behind a magnifying glass and a String Match Alarm Set icon has multiple sheets of paper behind a magnifying glass.

If an Application has generated no Alarms, then a special Alarm called an "OkAlarm" (illustrated in Figure 4.6) will be displayed that indicates that the Application has no unresolved Alarms.

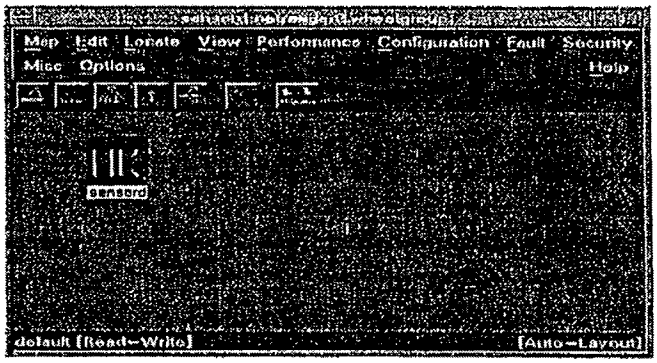


Figure 4.6: An OkAlarm

Adding Entities

In general, there are four types of icon symbols: alarms (which include Alarm Sets and OkAlarms), applications, machines, and collections.

Alarm symbols, at the bottom of the submap hierarchy, can only be created by the *nrdirmap* application. An alarm symbol is created whenever an event that exceeds a user-defined threshold is received. There is no way for a user to manually create an alarm symbol.

There are two ways that Application and Machine symbols can be created. First, if an application or host from which an event emanates is not already represented in the map, then *nrdirmap* will create the symbols for you.

If you do not want to wait until an alarm comes in to have a machine or an application represented in a map, you can add the symbols manually. The next two sections describe how to add machines and applications.

Manually Adding an NSX Machine Symbol

To manually add an NSX machine symbol, follow these steps:

1. **Double-click on a Collection symbol to open the Collection submap (the symbol on the root submap labeled "NetRanger" is a Collection symbol). Machines can only be added to Collection submaps! Do not try to add a Machine to a non-Collection submap.**

2. **Select the Edit→Add Object menu function. The Add Object Palette willw appear.**

NetView Users: ***Select Edit→Add→Object instead.***

---

3. **Click on the Net Device icon. Several Icons will appear in the bottom of the palette (see Figure 4.7).**

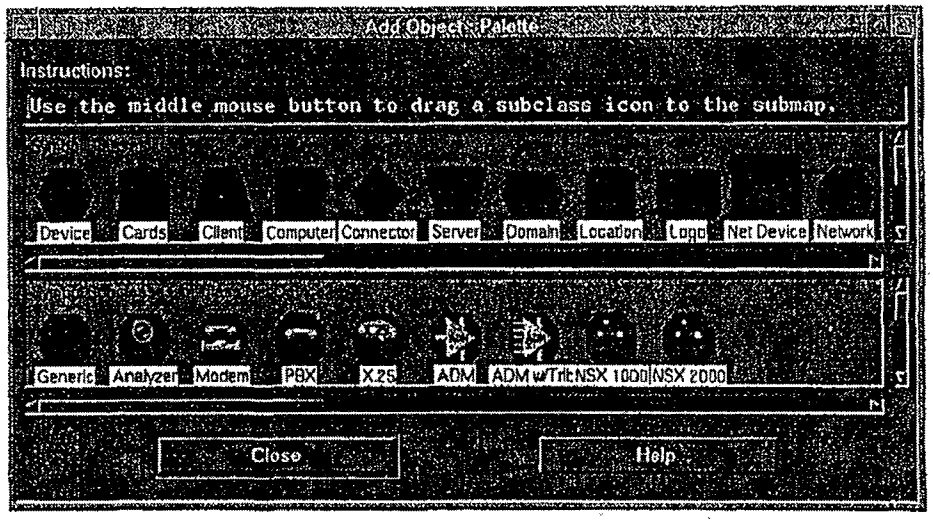


Figure 4.7: The Add Object Palette

4. **Position the mouse pointer over the NSX 2000 icon, press and hold the middle mouse button, and drag the NSX 2000 icon to the collection submap. An Add Object window should appear.**

5. **Select NetRanger/Director from the list, and press the Set Object Attributes button.**

6. **In the hostname field, enter the name of the NSX machine exactly as you entered it in the /usr/nr/etc/hosts file.**

7. **Press the Verify button. If you entered the hostname correctly, NetRanger/ Director will populate the Organization and Host ID fields for you.**

8. **Once the hostname, Organization ID, and Host ID are correct, press OK.**

9. **Press the Set Selection Name button. Choose the selection name from the list and then press OK in the Selection Name window. Press the OK button in the Add Object Window.**

NetView Users: Please skip step #9.

Manually Adding an Application Symbol

1. **Double-click on the NSX Machine to which you wish to add the application. Applications can only be added to Machine submaps! Do not try to add an Application to a non-machine submap!**

2. **If the Add Object Palette is not already displayed, bring it up by selecting the Edit→Add Object menu function.**

NetView Users: Select Edit→Add→Object instead.

---

3. **From the Add Object Palette, click on the WGC Application icon. Several icons should appear in the bottom of the window.**

4. **Using the same technique described above, drag the application icon to the NSX submap.**

5. **Select NetRanger Director from the list, and press the Set Object Attributes button.**

6. **All fields will be populated for you. Press OK.**

7. **Press the Set Selection Name button. Choose the selection name from the list and then press OK in the Selection Name window. Press the OK button in the Add Object Window. You should see the Application icon turn green, because a green OkAlarm will be created automatically in the submap of the Application you just added.**

NetView Users: Please skip step #7.

Manually Adding the Director Machine

Normally, you will not need to manually add the Director Machine. The Director Machine symbol is added for you automatically when nrdirmap comes up the first time. However, it is possible to manually delete the Director Machine, and you may want to manually add the Director Machine back to the map at a later time. Also, if you ever change the Organization ID or Host ID of the Director, then you must delete the Director Machine symbol and add it back with the correct IDs.

NOTE

Only one Director Machine can be represented at a time. In a future release, this restriction may be eased, but for this release, you can only have one Director icon on a map.

The procedure for adding a Director Machine symbol is almost identical to the procedure for adding an NSX Machine symbol. The only differences are as follows:

- From the Object Palette, instead of clicking on the "Net Device" symbol class, click on the "Computer" symbol class.
- Instead of dragging an NSX Machine symbol from the palette, drag the Director Machine icon.
- When you press the Set Object Attributes button, you shouldn't have to enter any data. Unless configuration files (like /usr/nr/etc/hosts) are missing or incorrect on your Director Machine, nrdirmap should be able to fill in this information for you.

Manually Adding an NSX Collection

The **Top-Level NSX Collection** (the entity that appears on the root submap labeled **NetRanger**) is created for you. This is the only Collection that can appear on the root submap. Do *not* try to create additional Collections on the root submap.

NSX Collections are used to customize, or partition, the map. NSX Collections are good tools to use for grouping machines into logical units. See the section in this chapter entitled *How to Customize a Map* for more information about specific uses of NSX Collections.

Follow these steps to add an NSX Collection:

1. **Double-click on the NSX Collection's submap to which you want to add the new NSX Collection. NSX Collections can only be added to NSX Collection submaps! Do not try to add an NSX Collection to a non-NSX Collection submap!**

2. **If the Add Object Palette is not already displayed, bring it up by selecting the Edit→Add Object menu function.**

3. **From the Add Object Palette, click on the Location icon. Several icons will appear in the bottom of the window.**

4. **Using the same technique described above, drag the symbol containing the WheelGroup Logo to the NSX submap.**

5. **Select NetRanger Director from the list, and press the Set Object Attributes button.**

6. **In the NSX Collection Name field, enter the name of the NSX Collection you just moved to the submap. This name can be any unique string. For example, "New York," "Building 162," or "10.1.1 Machines" would be legitimate NSX Collection Names.**

7. **Press the Verify button.**

8. **Press OK.**

9. **Press the Set Selection Name button. Choose the selection name from the list and then press OK in the Selection Name window. Press the OK button in the Add Object Window.**

NetView Users: Please skip step #9.

Modifying Entity Attributes

Once an entity has been created (either by a user or by nrdirmap), it is often helpful to view a listing of the entity's attributes. Some attributes can be edited by the user (for instance, a machine's Point of Contact), and other attributes are read-only (for instance, an alarm's Date). OpenView/NetView provides a visual indication of which fields are changeable and which are not.

To display an entity's attributes in OpenView, select the entity with the mouse pointer, and then select the menu function **Edit→Describe/Modify Object**. You can also select the entity and type Ctrl+O, or you can put the mouse pointer over the icon, press the right mouse button, and select **Describe/Modify Object**. A pop-up window will appear. Select **NetRanger Director** from the list of applications, and then press the **Configure** button.

NetView Users: ***To display an entity's attributes, select the entity with the mouse pointer and then select the menu function Edit→Modify/Describe→Object. You can also select the entity and type Ctrl + O. A pop-up window will appear. Select NetRanger Director from the list of applications and press the Configure button.***

Different entities have different attributes, so each entity will be discussed separately.

NSX Collection Attributes

The NSX Collection Name is the single attribute of an NSX Collection. If you change the Name, and then press **OK** on the appropriate screens, the NSX Collection's symbol label and submap name will change to reflect the new name.

Machine Attributes

Machines have four attributes: **Organization ID**, **Host ID**, **Hostname**, and **Point Of Contact**. The **Hostname** and **Point of Contact** are editable, but the **Org ID** and **Host ID** are not. If you need to change an Org or Host ID, the best thing to do is to delete the machine and then re-add the machine with the correct IDs. If the hostname is changed, the Machine's symbol label will be the part of the hostname up to the first dot (.), and the submap name will be the entire hostname.

If you want to store more information about the Point of Contact than the single field will contain, there are two things you can do. First, you can use the **Object Comments** field to store the additional point of contact information. Second, you could put the point of contact information in a separate trouble-ticketing system.

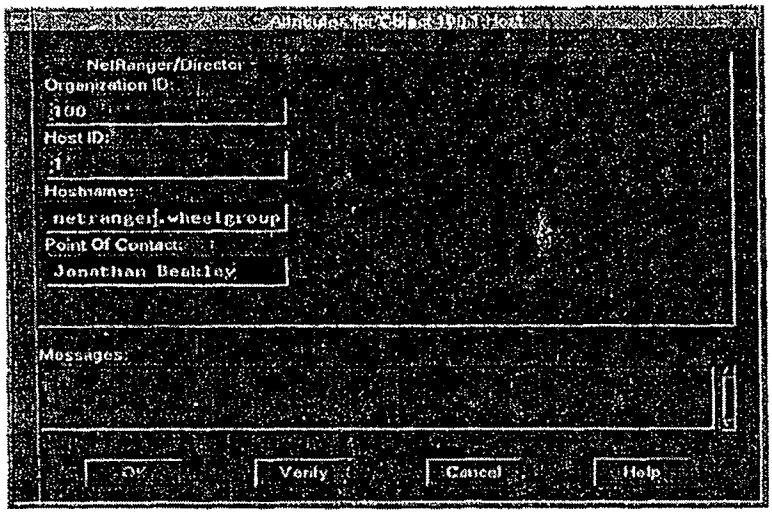


Figure 4.8: Object Attributes Window

Application Attributes

Applications have seven attributes: **Application Name, Minimum Marginal Status Severity, Minimum Critical Status Severity, Alarm Consolidation Threshold, Organization ID, Host ID,** and **Application ID**. The application name, status severity fields, and consolidation threshold are editable, but the ID fields are not. If you need to change an ID, delete the application and then re-add it with the correct IDs. If the application name is changed, the Application's symbol label will change to match the new application name, and the Application's submap name will reflect the new name, too (the format for the Application submap name is **<hostname>:<application name>**).

The **Minimum Marginal Status Severity** describes the lowest severity status an event can have before a marginal (yellow) alarm is created to represent that event. For example, if the minimum marginal status severity is 3, and a severity 2 alarm comes in, then no alarm entity will be created.

NOTE

The higher the severity level, the more severe the alarm. Currently, severity 5 is the highest severity level assigned by the *sensord* daemon.

The **Minimum Critical Status Severity** describes the lowest severity an event can have before a critical (red) alarm is created to represent that event. For example, if the minimum critical status severity is 3, and a severity 4 alarm comes in, then a critical alarm will be created.

NOTE

If you change a status severity value, only events generated **after the change** will be affected. If you increase a threshold severity level from 2 to 3, *nrdirmap* will not remove any existing level 2 alarms from the application's submap. Also, if you decrease a threshold severity level from 3 to 2, nrdirmap will not check historical log files and create alarm icons for severity level 2s that may have occurred in the past. Note that Connections and Alarms do not have submaps.

The **Alarm Consolidation Threshold** describes how many identical alarms must be received before the alarms are replaced by a single "Alarm Set" icon. By default, if two or more alarms are received that are alike in all respects except for timestamp and sequence number, nrdirmap will represent these alarms with a single "Alarm Set" icon. This prevents the screen from being cluttered when many similar alarms are received.

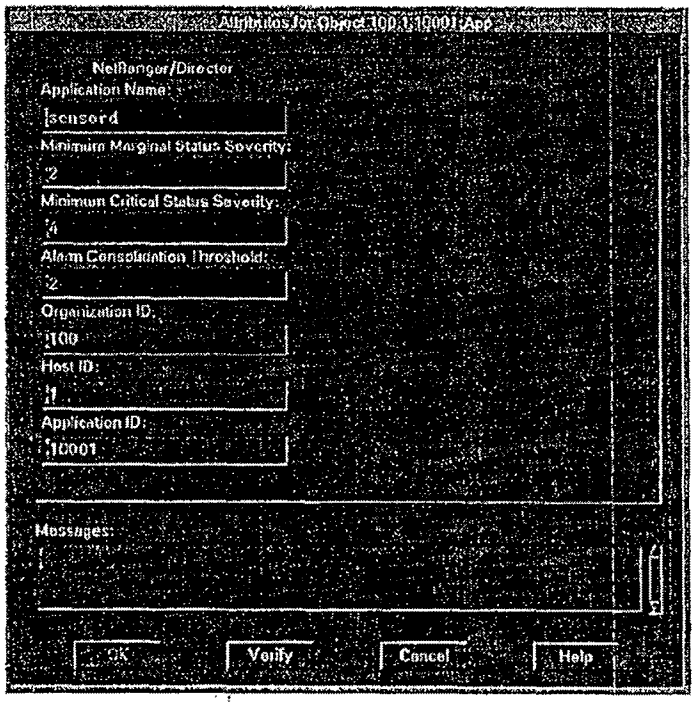


Figure 4.9: sensord Attribute Information

Alarm Attributes

Alarms have many attributes: **Name, Severity, Source Port, Destination Port, Source Address, Destination Address, Router Address, Date, Is Source Address Protected, Is Destination Address Protected, Details, Signature ID, Subsignature ID, Organization ID, Host ID, Application ID, and Instance ID.** All alarm attributes are read-only.

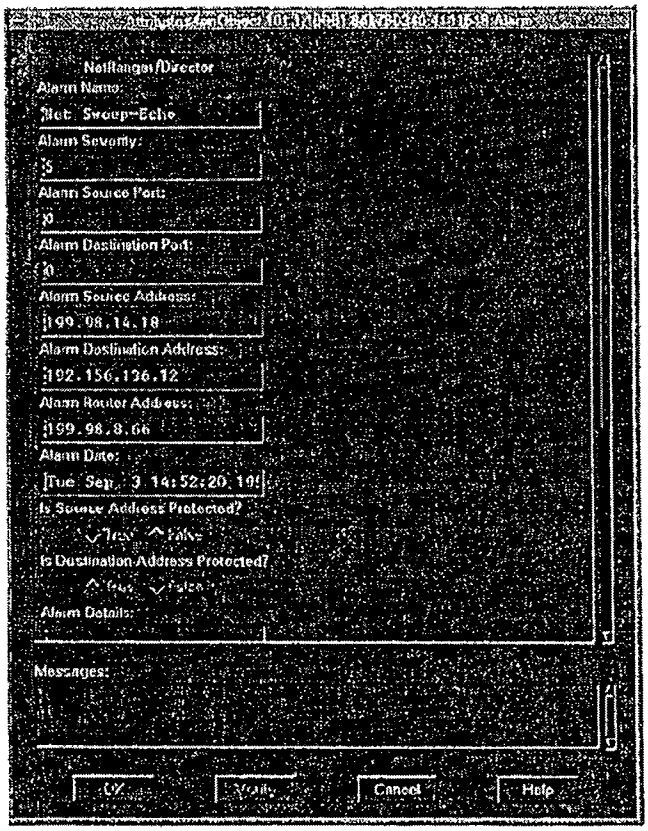


Figure 4.10: Alarm Event Attributes

By default, when two or more alarms are received that are alike in all respects except for timestamp and sequence number, nrdirmap will represent these alarms with a single "Alarm Set" icon. The attributes of an Alarm Set are almost the same as the normal Alarm. The Alarm Set does not have a timestamp and sequence number. Instead, it has an Alarm Count, Date of First Alarm in Set, and Date of Last Alarm in Set.

Once an Alarm Set is created, if additional matching alarms are received, the Alarm Count is incremented, and the Alarm Date(s) are changed if applicable. Note that the symbol label for an Alarm Set is similar to an Alarm, except that after the Alarm Name, the Alarm Count is given.

The special **OkAlarm** that indicates that an application has no unresolved alarms has only four attributes: **Date**, **Organization ID**, **Host ID**, and **Application ID**. All these fields are read-only. The Date field specifies the time at which the **OkAlarm** was created. This gives a lower boundary to the last time that the application in question detected an attack.

Deleting Entities

When you want to remove a symbol (and its corresponding database object), you must select the symbol and then choose the **Edit→Delete→From *All* Submaps** menu option. The most common usage of the delete function is deleting an alarm symbol once the potential hacking attempt has been diagnosed and resolved.

NetView Users: Select Edit→Delete Object→From All Submaps instead.

There are rules governing the deletion of symbols that help prevent the accidental removal of alarms and other symbols.

NOTE

Applications and Machines can NOT be deleted until ALL of their alarms have been deleted.

This forces the user to go into the submap containing the alarms and specify that it is OK to delete the alarms. This helps prevent a hacking attempt from going unnoticed.

Once an application or host has had all of its alarms resolved (and deleted), you are free to delete the application or machine.

NOTE

If you delete an application or machine, and then an event is received for that machine, the machine will be redrawn on the map. In a case like this, it might be better to unmanage the machine (see the description of the Unmanage function discussed later in this chapter).

Because it would be very easy to accidentally delete large groups of machines, non-empty Collections cannot be deleted. If you have a Collection that contains many machines, and you want to delete the Collection, you must first go into the Collection submap and delete all of the machines (and of course, the machines must have their alarms deleted before the machines can be deleted). Once you have emptied the collection submap, you can then delete the Collection.

NOTE

Never use the Delete Submap function! *nrdirmap* does *not* support this function. Always use the **Delete Object** function to delete entities!

How to Partition a Map

NSX Collection entities can be used to customize, or partition, a map. If the number of NSX machines you are monitoring is too great to represent on a single submap (for instance, the Top-Level NSX Collection submap), you can create additional Collections, and then add Machine icons to those Collection submaps. This allows you to create a hierarchical grouping of machines.

For example, if you had 25 NSX machines in Los Angeles, and 35 machines in New York, you could create an "LA Collection" entity and an "NY NSX Collection" entity. You could then add the NY NSX Machines to the NY Collection, and then add the LA NSX Machines to the LA Collection. This allows you to have fewer symbols per submap, which makes locating symbols and diagnosing problems faster and easier.

NOTE

To put a machine in a collection, you must use the **Add Object** function. If a machine is already represented on the map, and you want to move the representation (the symbol) from one collection to another, you must delete the machine and then re-add it. ***nrdirmap* does *not* support the "Cut and Paste" functionality!** Use of the Cut and Paste functionality on *nrdirmap* entities will yield unexpected results. *You must delete a machine and then re-add it to move the machine symbol from one Collection to another.*

Changing Map Configuration Parameters

There are five global Map-level configuration parameters that can be set. To see these parameters select the menu option **Map→Maps→Describe/Modify**.

NetView Users: ***Select File→Describe Map.***

You will then see a pop-up window. On this window, choose the **NetRanger Director** application, and then press the **Configure** button.

A window with five parameters will appear. You will see the following questions:

1. **Default lowest event severity that generates marginal icon?**
2. **Default lowest event severity that generates critical icon?**
3. **Default Number of Identical Alarms before Icon Consolidation?**
4. **Should nrdirmap be enabled for this map?**
5. **Should new security alarms be shown on the IP Map?**

The answer to the first question specifies the minimum severity an event must have before a marginal (yellow) Alarm is generated. For instance, if you set this value to 2, then if any new applications are created, these applications will have marginal alarms generated for events whose status is two and higher. Of course, if you manually reconfigure the Application symbol to have a new marginal status threshold, then this default value will be overridden.

The answer to the second question specifies the minimum severity an event must have before a critical (red) Alarm is generated. For instance, if you set this value to 3, then if any new applications are created, these applications will have critical alarms generated for events whose status is three and higher. Of course, if you manually reconfigure the Application symbol to have a new critical status threshold, then this default value will be overridden.

The answer to the third question specifies the number of similar alarms that must be received before the alarms are replaced with an "Alarm Set". For instance, if you set this value to 3, then if three identical alarms (identical in all respects except for sequence number and timestamp) are received, then these 3 alarm icons will be replaced with a single Alarm Set whose properties match those of the alarms that were replaced. This function limits screen clutter by reducing the number of icons on the screen.

The answer to the fourth question tells whether nrdirmap is enabled for the currently opened map. By default, all maps will have nrdirmap enabled. If nrdirmap is disabled for a map, then no NetRanger security information will be displayed on that map. This function can be used to control access to NetRanger information.

NOTE

This parameter can only be set when a map is created. Once a map is created, this parameter cannot be changed. For more information on this option, see the section entitled Limiting Access to Security Information.

The fifth question asks if you want alarm icons to be drawn on the IPMap. *This option has no effect with this release. It may be used in a future version.* The IPMap is the submap hierarchy created by the ipmap application. The ipmap application is the part of OpenView/NetView that draws a picture of the IP Topology. The advantage of having alarms represented on the IPMap is that you can view Fault status and Security status on the same screen. The disadvantage is that performance is degraded because extra icons are being created.

NetRanger Director User Interface Menu Functions

On the main menu bar, the menu option called **Security** contains many useful NetRanger Director functions. These functions are described below.

Remotely Configuring NetRanger Daemons

There are two ways to change the configuration of NetRanger daemons running on remote NSX machines.

1. **You can use the nrget/nrset infrastructure to modify daemon characteristics based on "tokens" that are specified by the user. These commands can be run from the command line of the Director machine.**

2. **You can also use the graphical browser utility that comes with the Director. To bring up the utility, select one or more Application or Machine symbols, and then select the menu option Security→Configure.**

   To learn more about nrConfigure, see the nrConfigure section on page 48.

Viewing Alarm Context Information

The **Security→Show→Context** menu function can be used to display data that was being transmitted across your network at the time a security event occurred. Context information is not available for all Alarm types. Currently, only Alarms that involve string matching contain context information.

The following signature IDs correspond to string matching alarms:

Signature ID	Type of string matching
3100-3104	e-mail
3200-3201	WWW
8000	General string matching

All Alarms that have the signature IDs shown above will be represented with "Alarm:Content" icons (icons containing a picture of a magnifying glass over a sheet of paper). These icons differentiate "Content" Alarms from "Intrusion" Alarms (which are represented by icons that contain pictures of lightning bolts).

To use the Show Context function, select one or more Alarm or Alarm Set icon(s), and choose **Security→Show→Context**. An "xnmappmon" window will appear that displays three fields:

- "String Matched". This field displays the string that was matched. The maximum length of this field is 64 bytes.
- "Context Buffer 1". This field displays up to 256 bytes of information that was transmitted in a single direction (either from or to the Server) at the time the string match occurred.
- "Context Buffer 2". This field displays up to 256 bytes of information that was transmitted in the opposite direction (either to or from the Server) at the time the string match occurred.

NOTE

All non-printable ASCII characters are displayed using a "\" character and two hex digits. For example, <ctrl-g>, which has ASCII value "07" in hex, is represented as "\07". The ASCII character "\" itself is represented as "\\".

If there is no context information available for an Alarm, then the "xnmappmon" window will display the following message:

```
Could not find context alarm information for alarm <Alarm Name>.
```

Viewing Event Lists

To view an ASCII list of the latest events that have been generated for a given application or machine, simply select either an Application or a Machine symbol from the map, and then choose the menu option **Security→Show→Current Events**.

This will execute a program that parses the log files in /usr/nr/var, looking for *all* events for the entity selected. Please note that this will include events that may be below the threshold for creating alarms.

Also note that this window is dynamically updated as new events come in. This is why the "hourglass" mouse pointer never goes away. The program does not stop until you press the **Stop** button, because it is always looking for new events.

To stop the search for new events, press the **Stop** button. After you have done this, you can enter new IDs (org/host ID pairs, or org/host/app ID tuples) and restart the search with the **Restart** button. You can also use the various save and print utilities to store the data you have collected.

Press **Stop** and then **Close** to stop the event search and close the window.

The events are displayed with an OpenView/NetView utility called **xnmappmon** (X-node Manager Application Monitor). You can change the fonts and layout of this utility by changing the application defaults file for this utility (see your network management platform documentation for details).

Viewing Database Status and Configuration

The **Security→Show→DB Info** function can be used to show information about the status and configuration of the NetRanger logging and database staging infrastructure.

To use this function, click on a Machine Icon that represents a machine running *sapd*, or click on a *sapd* Application Icon, and then select **Security→Show→DB Info**. An xnmappmon window will appear with three headings.

Under the heading

```
/usr/nr/var status for <orgId>.<hostId>.<appId>:
```

you will see information about the number and size of files in /usr/nr/var.

Under the heading

```
Run-time history for <orgId>.<hostId>.<appId>:
```

you will see information about the number of database processes that have been run.

Under the heading

```
Trigger configuration for <orgId>.<hostId>.<appId>:
```

you will see database configuration information in the format below.

In the following example, the ORACLE_LOAD trigger is configured such that the DBLoad process will run if the number of DIRFILES in /usr/nr/var/new is greater than or equal to 1.

```
ORACLE_LOAD  DIRFILES  1   /usr/nr/var/new  DBLoad
```

Resolving an Alarm's IP Addresses

The **Security→Show→Names** function can be used to find the hostnames of an alarm's source and destination IP addresses. To use this function, select one or more Alarms (or Alarm Sets), and select **Security→Show→Names**.

An xnmappmon window will display the hostnames if they can be found. If the IP addresses cannot be resolved, you will see the following message:

```
"*** <Resolver> can't find <IP Address>: Non-existent domain"
```

Determining the Version of a Remote NetRanger Daemon

The Director includes a utility called *nrVersion* that determines the Version of NetRanger code running on a machine. This function is helpful when diagnosing problems or upgrading software. To run nrVersion, simply select one or more Machine or Application symbol, and then choose **Security→Show→Version**.

An "xnmappmon" window will display the version numbers of all NetRanger applications that are selected, and/or the version numbers of all NetRanger applications on any machines that are selected.

If you see the following message:

```
Error: Problem sending query to nr.configd
```

then the version of nr.configd that you are using does not support this menu function. You should also ensure that nr.configd is running on the remote machine.

Shunning IP Addresses and Class C IP Networks

The **Security→Shun** functions can be used to manually shun (block) incoming IP traffic. To shun traffic that emanates from an Alarm's Source IP Address, select the Alarm (or Alarm Set), and choose **Security→Shun→Source IP**. To shun traffic that emanates from any IP address within the Class C Network that contains an Alarm's Source IP Address, select the Alarm (or Alarm Set), and select **Security→Shun→Source Net**.

In either case, an xnmappmon window will display the output of the nrexec command that was used to shun the traffic.

Please note that the default timeout value for the nrexec command is 10 seconds, and that the default duration for the shun is 1440 minutes (one day). To extend the duration, to stop the shunning, or to otherwise exercise more granular control, use the **Security→Configure** menu function.

Unshunning IP Addresses and Class C IP Networks

The **Security→Unshun** functions can be used to manually unshun (allow the transmission of) incoming IP traffic. These functions are used to "undo" the effects of the **Security→Shun** functions.

To unshun traffic that emanates from an Alarm's Source IP Address, select the Alarm (or Alarm Set), and choose **Security→Unshun→Source IP**. To unshun traffic that emanates from any IP address within the Class C Network that contains an Alarm's Source IP Address, select the Alarm (or Alarm Set), and select **Security→Unshun→Source Net**.

In either case, an xnmappmon window will display the output of the nrexec command that was used to unshun the traffic.

Please note that if traffic is not already shunned for the selected Alarm, then the Unshun action will have no effect.

Saving Object Data to a File

Use the **Security→Save To File** function to direct the attributes of one or more objects to a file. This is helpful if you want to send an e-mail message about alarm details to someone.

To use this function, select one or more symbols on the map, and then choose **Security→Save To File**. An ASCII file will be created in /usr/nr/tmp. The filename(s) will match the selection name(s) of the object(s) you selected.

This function uses the OpenView/NetView ovobjprint utility. For more information about ovobjprint, see the ovobjprint man page.

Finding Out About nrdirmap

Use the **Security→About** function to find information about the version of nrdirmap that you are using.

Creating a Trouble Ticket

If you have the Remedy ARS (server or client) installed and configured on your Director machine, then you can use the **Security→Trouble Ticket** menu option to create a trouble ticket. If you do not have the Remedy ARS installed, then this menu function will have no effect.

To create a Remedy Trouble ticket, select one or more Alarm or Alarm Set icons and choose the **Security→Trouble Ticket** menu option. One trouble ticket will be created for each icon that is selected. Please note that this means that only one ticket will be created for a selected Alarm Set, even though the Alarm Set represents multiple Alarm notifications. In addition, the date timestamp within the ticket will be the timestamp of the most recent event in the Alarm Set. This function uses NetRanger *sapx* (Security Analysis Package eXtractor) infrastructure to load the data into the Remedy schema. Refer to Chapter 5 in this *User's Guide* to learn how to configure Remedy to work with NetRanger.

Changing IP Addresses, Hostnames, and NetRanger IDs

If you change the IP characteristics of either a Director or NSX machine, or if you change the NetRanger communication infrastructure characteristics (like hostID, orgID, host name, and organization name), you *must* ensure that the appropriate configuration files have been changed on *all* your NetRanger machines, including the Director machine.

If you change the hostID or organizationID of either an NSX machine or the Director machine, after making the necessary changes to the configuration files, ensure that you use the **Edit→Delete** menu option to delete the machine from the map. After this is done, use the **Edit→Add Object** menu function to add the machine back to the map with the proper ids. See the sections on adding and deleting objects for more information about these procedures.

If this changes...	ensure that it is changed here, too (use nrconfig):
an IP Address	/usr/nr/etc/routes
	/usr/nr/etc/sensord.conf
	/usr/nr/etc/managed.conf
host or organization names	/usr/nr/etc/destinations
	/usr/nr/etc/auths
	/usr/nr/etc/hosts
	/usr/nr/etc/routes
	/usr/nr/etc/smid.conf
host or organization IDs	/usr/nr/etc/hosts

If the IP address or hostname of the network management station must be changed, consult your network management documentation to learn about what configuration changes must be made to your network management platform. On HP systems, it is recommended that you shut down the user interface, stop the OpenView daemons, stop the NetRanger daemons, and then use sam to reconfigure the IP information. If you are changing the hostname, you should run `/etc/set_parms hostname` to ensure that the Common Desktop Environment is aware of the new hostname. Once this is done, and once any additional OpenView-specific configuration is complete (as specified in the OpenView documentation), it is recommended that you reboot your machine.

Changing Registration Files

All OVw Applications have a configuration file called a Registration File that tells the User Interface (OVw) how to treat the application. On HP systems, registration files are kept in `$OV_REGISTRATION/C`.

In general, registration files should not be modified, but there are a few circumstances in which it is helpful to edit registration files. Registration files contain the command that is actually used to launch the OVw Application, so registration files are good places to edit an OVw Application's command-line parameters.

The following table lists nrdirmap's command line parameters:

Opt	Params	Function
-a	<integer>	default Alarm consolidation threshold for new maps
-c	<integer>	default Critical value threshold for new maps
-d	<none>	Disable nrdirmap for new maps by default
-f	<none>	force Full synchronization
-k	<none>	Keep symbol copies during delete
-i	<integer>	number of seconds to be Idle before sleep
-m	<non-zero int>	default Marginal value threshold for new maps
-o	<integer>	maximum number of Objects represented in a map
-p	<none>	Propagate to IPMap for new maps
-s	<none>	Secure new map creation
-t	<none>	Tracing enabled

-a, Alarm consolidation threshold

By default, if two or more alarms are received that are alike in all respects except for timestamp and sequence number, nrdirmap will represent these alarms with a single "Alarm Set" icon. This prevents the screen from being cluttered when many similar alarms are received.

An alarm consolidation threshold is configurable for each application object that is represented in the map.

The -a option is used to define a new *default* alarm consolidation threshold. This default value is applied to all application objects that are created *after you change the default value*. For example, if all of the application icons in your map have an alarm consolidation of 2, and then you set the -a option to 5, then whenever a new application is created, the new application will have the threshold of 5. Please note that the -a option will have no effect on *existing* application entities.

The default is 2. A value of 0 means no alarm consolidation. Any integer zero or higher is valid.

-c, Critical value threshold

By default, if nrdirmap receives an event with a severity level of 4 or higher, the symbol that represents that alarm will have "Critical" status. Unless the user specifies otherwise, a symbol with Critical status will be red.

A critical value threshold is configurable for each application object that is represented in the map.

The -c option is used to define a new *default* critical value threshold. This default value is applied to all application objects that are created *after you change the default value.* For example, if all of the application icons in your map have a critical threshold of 4, and then you set the -c option to 3, then whenever a new application is created, the new application will have the threshold of 3. Please note that the -c option will have no effect on *existing* application entities.

-d, Disable nrdirmap by default for new maps

By default, when a new map is created, nrdirmap will be enabled, which means that nrdirmap will display security information on the new map. If you would like for nrdirmap to be disabled for new maps by default, add the "-d" option to the nrdirmap registration file.

See the section in this chapter entitled *Limiting Access to Security Information* for more information about enabling and disabling nrdirmap.

-f, force Full synchronization

By default, when a symbol that exists in multiple maps is deleted from a map, the symbol will not be redrawn on the map when the user interface is stopped and restarted. The idea is that once you delete a symbol from a map, the symbol is deleted permanently. Once a symbol is deleted from *all* maps, the object that the symbol represents is deleted from the object database.

This means that there could be objects in the database that might not be represented as symbols in one or more of your maps (for instance, if you have two maps, and you delete an alarm symbol from one of the maps, the alarm object is still in the database, but there is no symbol for that object in one of your maps).

If you ever want to "refresh" a map to ensure that all objects in the database are represented in a map, use the -f option. This will force nrdirmap to represent all objects in the database as symbols on the map you just brought up.

Note that this option should have no effect if you only have one map.

If you have multiple maps, and you want to "recover" a symbol that was accidentally deleted from a map, you can use this option (assuming that the object is still represented as a symbol in another map).

-i, seconds to be idle before sleep

By default, if nrdirmap does not receive an alarm or a user interface callback for 5 seconds, it will go to "sleep", and wake up once per second to check for new events. Once a new event is received, it will handle new events as fast as they come in, until 5 seconds pass with no more events. At this point, nrdirmap goes back to sleep.

To change the amount of time in seconds before sleep, use this option. Under normal circumstances, there will be no reason to change this value from the default of 5 seconds.

-k, Keep symbol copies during delete

By default, when you delete an entity (for instance, a machine, application, or alarm) from a submap in the Director submap hierarchy, *all* symbols for the entity will be removed from the map. This means that if you make a copy of a symbol and then delete the original symbol, nrdirmap will ensure that the copy is deleted, too. If you want to keep copied symbols on the map when you delete a NetRanger symbol, set the -k option.

WheelGroup recommends the default setting unless you have a specific reason to create and retain symbol copies.

-m, Marginal value threshold

By default, if nrdirmap receives an event with a severity level greater than or equal to 3 and less than the critical threshold value (discussed above), the symbol that represents that alarm will have "Marginal" status. Unless the user specifies otherwise, a symbol with Marginal status will be yellow.

A marginal value threshold is configurable for each application object that is represented in the map.

The -m option is used to define a new *default* marginal value threshold. This default value is applied to all application objects that are created *after you change the default value.* For example, if all of the application icons in your map have a critical threshold of 3, and then you set the -m option to 2, then whenever a new application is created, the new application will have the threshold of 2. Please note that the -m option will have no effect on *existing* application entities.

-o, maximum number of Objects represented in a map

The nrdirmap application creates an object in the OpenView Windows Object Database (ovwdb) for every Alarm (or Alarm Set) icon represented in a map. If the number of objects in the Object Database grows beyond a certain level, OpenView performance can deteriorate and in extreme cases, OpenView can fail.

To prevent these problems, nrdirmap by default will represent no more than 10,000 entities (alarms, machines, applications, etc.) in a map at a given time and no more than 1,000 icons on a given submap (window). If a new event comes in that would cause nrdirmap to exceed either of these threshold values, then nrdirmap will print a warning message to standard out and then redirect the alarm data to a buffer file in /usr/nr/var for safekeeping, thus ensuring the integrity of your OpenView data.

To pull the buffer data from the file into the OpenView user interface, delete enough alarms from the user interface (using the Delete Object menu function) to reduce the number of icons and database objects and then bring the user interface down and back up again. nrdirmap will read the file and create the new alarm icons.

If you are routinely receiving tens of thousands of severe alarms in short periods of time from a given application, you should consider redefining the security policy on your NSX, or increase the Minimum Marginal Status Severity level for that application object.

High-end workstations with plenty of RAM may be able to handle more than 10,000 Objects in a map. If you want to increase (or decrease) this maximum number of Objects value, add the text `-o <value>` to the nrdirmap registration file. It is not recommended that this value exceed 50,000. If you increase this value, you may wish to increase ovwdb's cache size by editing the $OV_LRF/ovwdb.lrf file to improve ovwdb's performance (see the appropriate OpenView documentation for more information).

The 1,000 icons per submap (window) limit is not configurable because this limitation is based on the number of pixels that can be devoted to a window, rather than a value that is user-configurable (such as RAM, disk space, etc.).

-p, Propagate to IP Map

This option currently has no effect. This option may be used in a later version of the product.

-s, Secure map creation

Use this option to ensure that only authorized users can create maps that contain NetRanger information. When the "-s" option is enabled, maps with nrdirmap enabled can only be created from maps that already have nrdirmap enabled. This ensures that a user who does not have access to nrdirmap-enabled maps (and therefore, does not have access to security information) will not be able to create a map that has nrdirmap enabled.

See the section in this chapter entitled Limiting Access to NetRanger Director Security Information on page 44 for more information about enabling and disabling nrdirmap.

NOTE

Enabling the "-s" option will preclude creating nrdirmap-enabled maps from the command line using the `ovw -m <map name>` command. Once the "-s" option is enabled, the only way to create a new map with nrdirmap enabled is to open a map that already has nrdirmap enabled, and then use the **Map→Maps→New** menu function.

NetView Users: Select File→New Map instead.

-t, Tracing enabled

If nrdirmap is malfunctioning, your authorized service representative might instruct you to enable tracing by adding the -t option. After you set this option, it is best to bring down the user interface, and then bring it back up by typing the following command:

```
ovw > /usr/nr/tmp/nrdirmap.out
```

NOTE

This will create a file called nrdirmap.out with information that can be used by a WheelGroup representative to diagnose the problem. Please note that if you do not redirect the trace messages, the messages will go to standard out.

Command Line Parameter Examples

To enable tracing, type:	`Command -Shared -Initial -Restart "nrdirmap -t";`
To create marginal icons for level 2 alarms and critical icons for level 3 and higher alarms, type:	`Command -Shared -Initial -Restart "nrdirmap -m 2 -c 3";`
To enable tracing and to create Alarm Sets once 15 similar alarms are received, type:	`Command -Shared -Initial -Restart "nrdirmap -t -a 15";`

Changing the Number of Events Displayed in Event List

When you select a Machine or Application symbol and select the menu option **Security→Show Current Events**, by default the last 100 events associated with that entity are displayed (if less than 100 events are known, then all of the events are displayed). To change the number of events that are displayed, use an editor to modify the nrdirmap file, which is stored in `$OV_REGISTRATION/C` on HP and Sun systems, and `/usr/OV/registration/C` on IBM systems.

Replace the "100" with the number of your choice on the line shown in bold.

```
Action show_events  {
        SelectionRule (isWheelGroup && (isMachine || isApplication));
        MinSelected  1;
        Command "sh -c 'unset OVwSessionLoc \;
                $OV_BIN/xnmappmon \
                -selectList \" $(OVwSelections) \" \
                -commandTitle \" Show Current Events for \" \
                -appendSelectList \
                -appendSelectListToTitle \
                -multipleDialogs \
                -headingLine 2 \
                -geometry 900x600 \
                -followOutput \
                -unbufIO \
                -stopSignal 9 \
                -cmd /usr/nr/bin/filterLogByHostApp -l 100' ";
    }
```

Changing Symbol, Object, and Submap Characteristics

The OpenView user interface (ovw) provides functions that modify characteristics of symbols (icons) and database objects.

Unfortunately, the user interface is perhaps a bit *too* powerful, in that it allows a user to change data in ways that can confuse the nrdirmap application. This section describes what user modifications are allowed and which are not.

Object Modifications

Object Modifications are made through the **Edit→Describe/Modify Object** menu function.

NetView Users: ***Select Edit→Modify→Describe→Object.***

It is OK to change the Comments field, and it is OK to change any attribute field that is editable from the window that is accessed by selecting NetRanger Director and pressing the **View/Modify Object Attributes** button. Please note that any changes you make to an object *will apply to all maps.* Remember that symbols are map-specific, but database objects are shared among all maps.

You should not edit the Selection Name. The Selection Name field is a field that is used to uniquely identify an object.

Symbol Modifications

Symbol modifications are made through the **Describe→Modify Symbol** menu function.

NetView Users: ***Select Edit→Modify/Describe→Symbol.***

You can change the Display Label setting, and you can change a symbol from explodable to executable.

In general, changing the symbol label itself is not recommended, because nrdirmap has specific algorithms it uses to determine the symbol label, and it will override your label customization when the user interface is stopped and started.

You should not change the symbol status source because nrdirmap expects all symbols (except alarm symbols) to have Compound Status Source. This ensures that an alarm's status is always propagated upward in the submap hierarchy. Changing the status source will jeopardize this.

You should not change the symbol type. This will cause capability fields in the object database to be changed, and this will affect many different functions—including synchronization and callback communication between nrdirmap and ovw. Do not change a symbol's symbol type.

Submap Modifications

Submap modifications are made through the Map→Submap→Describe/Modify menu function.

NetView Users: Select Edit→Modify/Describe→Submap instead.

You can change the Comments and the Background Graphics fields. Changing the Submap Name is not recommended because, like the symbol label, the submap name is set by nrdirmap, and nrdirmap will override your customization when the user interface is shut down and brought back up.

Searching for Symbols

OpenView provides fairly powerful search utilities. These search utilities can be used to locate symbols that match certain criteria. The following three search functions might be useful when searching for alarm symbols:

- Locate by Object Attribute.
- Locate by Symbol Type.
- Locate by Symbol Status.

To use the Locate function, choose **Locate→Objects** from the main menu, and then pick the type of search you want.

For example, to view the number of unresolved **String Matches**, you could search by Symbol Type, and select the **Alarm:Content** symbol type. To determine how many critical elements you have in your network, you could do a search by Symbol Status, and then search for Critical (red) elements. Finally, to search for an alarm from a particular source IP address, you could search by Attribute, and then choose Source IP Address from the list of attributes, and then type in the source IP address you want to find.

Setting the Home Submap

When you start up the user interface, a submap that is designated the home submap is opened. By default, the root (top) level submap is the home submap. You may want to change the home submap to the child submap of the top level NSX Collection (the submap you get when you double-click on the NSX Collection that appears on the root level submap). If you want to make this change, follow these steps:

1. **Double-click the NetRanger icon.**
2. **Select the menu option Map→Submaps→Set This Submap As Home.**

NetView Users: ***Select Options→Set Home Submap.***

Changing a Submap Background

It is sometimes helpful to place a background picture on a submap to help identify the submap quickly. Submap backgrounds can also be used to help provide context for the different symbols on the submap (for instance, machine icons positioned strategically on a picture of a floor plan could help mark where the machines reside physically).

To add a submap background, open the submap that you want to change, and then select the menu option **Map→Submaps→Describe/Modify**.

NetView Users: ***Select Edit→Modify/Describe→Submap.***

Under the **Background Graphics:** heading, press the **Browse** button.

NetView Users: ***Press the Select button.***

From the pop-up list, select the background graphic of your choice. On HP systems, usastates.gif is a popular choice. You could also create a custom GIF file with any graphics program, and use that GIF file as a submap background. Press **OK** twice.

Repositioning Symbols on a Submap

You can use the mouse pointer to move symbols to different positions on a submap. However, if symbols are added to or removed from the submap, the user interface will automatically reposition *all* of the symbols on the submap, and the customization will be lost.

To prevent this, it is usually best to turn **automatic layout** off. To do this, choose **View→Automatic Layout**, and select the **off** option for either the current submap (if you are only repositioning symbols on a small subset of submaps) or for all submaps (if you reposition symbols frequently).

Hiding Symbols

Under some circumstances, you might want to prevent a symbol from appearing on a given submap, but you might not want, or be able, to delete the symbol. For instance, there could be a machine in a collection that you don't care about, but you can't delete it because it has unresolved alarms. Assume for the moment that there is some reason why you don't want to delete the alarms. In a situation like this, it is best to *hide* the symbols.

To hide a symbol, select it, then choose the menu option **Edit→Hide**.

NetView Users: ***Select Edit→Hide Objects.***

You are given a choice of hiding **This Submap** or **All Submaps**. Pick the option of your choice.

To "unhide" a symbol, simply select the **Edit→Show Hidden Objects** menu function.

Changing the Status Propagation Schemes

When a symbol has Compound Status Source, the status (color) of the symbol is based solely on the status of the symbol(s) in that symbol's child submap. OpenView provides the user with user-selectable sets of "rules" that the User Interface uses to determine the status of a symbol based on the status of the symbols in the child submap. These rulesets are called **Compound Status Source Propagation Schemes**, and the ruleset you choose will affect the color of the icons on the map.

To change the status propagation scheme, choose the **Map→Maps→Describe/Modify** menu option, select one of the radio buttons associated with a scheme, and then press **OK**.

NetView Users: Select File→Describe Map.

WheelGroup Corporation highly recommends that you select the Propagate Most Critical scheme.

Consult the documentation provided with your network management platform for more information about Compound Status Source.

Changing Appearances, Fonts, Window Sizes, Colors, Etc.

In OpenView there are special ASCII files called **Application Default** files that contain parameters that can be customized to change the look and feel of certain applications. To change fonts, window sizes, colors, etc. for the user interface in general, edit the OVw file which is in `$APP_DEFS` on OpenView systems and `/usr/OV/app-defaults` on NetView systems.

To modify the attributes of the various XNm applications, modify the XNm* files. The **Show Current Events** window uses an application called **xnmappmon** (X-Node Manager Application Monitor) to display data. If you want to change the appearance of this window, make the necessary modifications to this file.

Creating and Using Multiple Maps

The NetRanger Director supports the use of multiple OpenView Windows (ovw) maps. Using multiple maps in OpenView can be a little tricky, so consider reading the appropriate OpenView documentation before creating multiple maps.

Here are some things to remember about multiple maps:

- To delete an object from the object database, you must delete the object's symbols from all maps. For example, if an alarm object is represented on two different maps, the alarm must be deleted from both maps before the alarm is cleared from the object database.
- You can create customized maps for different users. Each user's map can have a different subset of NSX Machines displayed. This is helpful when trying to distribute responsibility for different NSX machines to different users.

To create a customized map for a user, either use the **Map→Maps→New** function or use the -m command-line parameter when invoking ovw (see the appropriate OpenView documentation for details).

NetView Users: ***Select File→New Map.***

When the new map is created, nrdirmap will represent all of the objects in the database on the new map. Once the new map has stopped Synchronizing, you can use the **Edit→Delete** function to remove the symbols that you don't want to be represented on that map.

NetView Users: ***Select Edit→Delete Object.***

Usually, Machine symbols are deleted to create a "user domain" with a subset of the configured NSX Machines.

If you delete a Machine from one map, but the Machine is represented in a second map, and if an alarm is generated by that Machine, then the alarm will only be represented on the map containing the Machine.

Please note that if you delete the Machine from *all* maps, then the machine object will be deleted from the database. If that Machine generates a new alarm, then a new database object will be created for that Machine, and the Machine will be represented in *all* maps again.

If you have an NSX Machine that sends alarms to the Director, but you don't want the NSX Machine to be represented in any maps, you should reconfigure the NSX machine to stop sending the alarms to the *smid* process running on the Director. This will prevent nrdirmap from receiving the events and creating the NSX Machine icon. If you still want the events from the NSX Machine to be logged in the Director machine, but not displayed by nrdirmap, configure the NSX Machine to send events to the *loggerd* daemon on the Director rather than the *smid* daemon.

If you want to put back symbols that you previously deleted, there are two things you can do if the objects that the symbols represented are still in the object database.

Option 1: You can use the **Edit→Add Object** menu function to add the object back to the map.

NetView Users: ***Select Edit→Add→Object.***

NOTE

You will get a few warning messages that you are adding an object that already exists. Press **OK** to continue adding the object.

nrdirmap only creates symbols for alarms that are received *after* the machine/application are added to the map. If you want to add a Machine or Application that already exists in the database and you want to also see what alarms already exist, you should use Option 2.

Option 2: If you have many objects that you want to add back to the map, it might be faster to bring the user interface down, and then add the -f option to the nrdirmap registration file. This will force nrdirmap to represent all objects in the database on the map. Refer to the section earlier in this chapter on editing registration files for more information on the -f option.

If one map is open (being viewed with ovw) and a second map is closed, and an event comes in, an alarm will be displayed on the open map. If the alarm is deleted from the open map before the second map is opened, then the alarm will be deleted from the database, and the alarm will not be represented on the second map when the second map is opened. On the other hand, if the second map is opened before the alarm is deleted from the first map, the alarm will be read from the database and will be represented in the second map.

Note that Object attributes apply to all maps. In other words, any time that the **Edit→Describe/Modify Object** option is used to edit an object, these changes will apply to all maps. Unfortunately, there is no way to customize Object attributes on a per-map basis.

NetView Users: ***Select Edit→Modify/Describe→Object.***

Using Read-Only Maps

The NetRanger Director supports the use of read-only user interface sessions. Using read-only maps in OpenView can be a little tricky, so consider reading the appropriate OpenView documentation before using read-only maps.

Here are some things to remember about read-only maps:

- It is not possible to add symbols to or remove symbols from a read-only map. This is a restriction in OpenView that cannot be circumvented. This means that it is impossible for nrdirmap to create an alarm symbol on a read-only map.
- As a result, when an event is received by a read-only map, nrdirmap will change the status of the machine's application's OkAlarm symbol from Normal (green) to the status of the event that was received. This means that even on a read-only map, you should see the status color propagation when an alarm is received.
- Once you see the OkAlarm (and therefore, the Application and Machine) change color on a read-only map, you should use the **Map→Refresh Map** menu option to view an updated copy of the map. This will allow you to view the alarms that have come in.

NetView Users: ***Select File→Refresh Map.***

- Please note that if you have no open read-only copies of the map, then refreshing the map will not provide you with additional information. You must have a read/write copy running to ensure that the map is updated properly.
- If a read-only nrdirmap receives an event, and if the application that generated the alarm does not have an OkAlarm displayed on that read-only map (for instance, if there are other alarms already displayed), then nrdirmap has no OkAlarm symbol to modify in order to reflect the new alarm. As a result, it is advised that users of read-only maps ensure that the maps contain OkAlarms so that status changes are seen immediately. If this is not possible, then read-only users should either refresh the maps occasionally or use *eventd* to notify the user that an event has been received and the map should be refreshed.

Limiting Access to NetRanger Director Security Information

There are functions in the NetRanger Director and in OpenView that can be used together to control which OpenView users have access to security management information.

If you have multiple users who have permissions to run the OpenView user interface, but if you want only a subset of those users to be able to view security information, read the section below.

Limiting access to security information is done by disabling the nrdirmap application for one or more OpenView maps. Once nrdirmap is disabled for a map, nrdirmap will not try to display security information on that map.

Once you have one or more maps that have nrdirmap disabled, and one or more maps that have nrdirmap enabled, you can set the permissions of the OpenView maps to limit which users can access which maps. The result is that only the users with permissions to open the nrdirmap-enabled maps will have the ability to view security information.

Disabling nrdirmap

nrdirmap is disabled on a per-map basis. You must decide whether or not to disable nrdirmap whenever you create a new map.

NOTE

Once a map is created and the decision has been made to enable or disable nrdirmap, the decision is permanent. For instance, once nrdirmap has been disabled for a map, it cannot be enabled.

Disabling nrdirmap can be done in two different ways, depending on how the map is created:

1. **If you create the map from the Map→Maps→New menu option, choose NetRanger Director from the list of configurable applications, and press the Configure button. You will see an option that reads**

   ```
   Should nrdirmap be enabled for this map?
   ```

   Choose `True` to enable nrdirmap and choose `False` to disable nrdirmap. The default is `False` if the -d option appears in the nrdirmap registration file; the default is `True` otherwise.

2. **If you create the map from the command line using the ovw -m option, nrdirmap will be disabled if the -d option appears in the nrdirmap registration file; nrdirmap will be enabled otherwise.**

Setting User Groups

All users who should have access to security information MUST be in the group netrangr. Furthermore, all users who should not have access to security information should NOT be in the group netrangr.

On HP systems, users can be added to and removed from groups using the SAM utility. On Sun systems, the admintool can be used.

NOTE

On HP Systems, if a user is in the group netrangr (because of the configuration of the /etc/group file), but the user's primary group is not netrangr (because the group ID listed in /etc/passwd for the user is not the group ID assigned to the group netrangr), then before executing nrdirmap, the user must type `newgrp - netrangr`. **Sun and IBM users do not have to do this.**

Setting Map Groups and Permissions

Once you have created your maps, you can use the `ovwchgrp` and `ovwchmod` commands to set map permissions.

NOTE

OpenView users should have $OV_BIN in their paths, and IBM users have /usr/OV/bin in their paths.

Maps that have nrdirmap enabled should be readable by users who have access to the /usr/nr subdirectory. This means that maps that have nrdirmap enabled should be owned by the group netrangr, and/or be owned by a user who is in the group netrangr.

Ensure that map permissions are set so that the nrdirmap-enabled maps are not readable by people not in the netrangr group. For example, to allow only the user netrangr and users in the group netrangr to read and write a map called "default", type

```
ovwchown netrangr default
ovwchgrp netrangr default
ovwchmod 660 default
```

Use the `$OV_BIN/ovwls` command to list the maps and verify the permissions.

In the following example, two maps are created. The map called "secinfo" has nrdirmap enabled, and it will be accessible only by the user angie and by other users in the group netrangr. The map called "nosecinfo" has nrdirmap disabled, and it will be accessible only by the user bob and users in the group "staff". Please note that the following example assumes that the -d option has not been added to the nrdirmap registration file.

Set-up

1. **Ensure that user "angie" is in the UNIX group "netrangr".**
2. **Ensure that user "bob" is not in the UNIX group "netrangr".**

Create the "secinfo" map

1. **Log in as user "angie".**
2. **Create the map by typing**

```
ovw -m secinfo &
```

Set the "secinfo" map permissions

1. **From the command line, type**

```
ovwchown angie secinfo
ovwchgrp netrangr secinfo
ovwchmod 660 secinfo
```

Create the "nosecinfo" map

1. **Choose the Map→Maps→New menu option.**
2. **Enter a map name, select "NetRanger/Director" from the list of configurable applications, and press the Configure button.**
3. **Choose `False` in response to the following question:**

```
Should nrdirmap be enabled for this map?
```

4. **Choose OK to create the map.**

Set the "nosecinfo" map permissions

1. **Type the following from the command line:**

```
ovwchown bob nosecinfo
ovwchgrp staff nosecinfo
ovwchmod 660 nosecinfo
```

Now, the map viewable by bob will not contain security information, but the map viewable by angie will.

nrConfigure

Overview

nrConfigure is a Java-based graphical user interface (GUI) that allows you to remotely configure NetRanger applications and access those configurations. It supports all the functionality of the individual nrget, nrgetbulk, nrset, nrunset, and nrexec commands. This GUI-based environment allows you to see an application's tokens, each token's actions, and each action's optional values. Figure 4.11 illustrates the nrConfigure window.

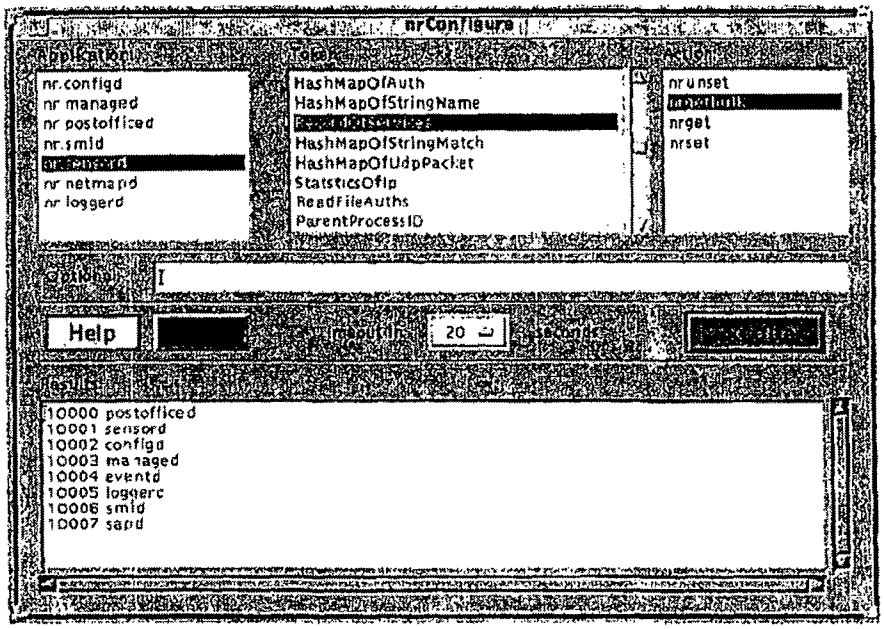


Figure 4.11: The nrConfigure Window

Architecture

The nrConfigure GUI works in the following manner. When you press **Execute** for a given application, token, and action, a packet is sent to *configd*, which sends it on to *postofficed*. *postofficed* sends it on to the proper daemon, which processes the requests and replies. The reply returns through *postofficed* to *configd* and appears in the nrConfigure's Results window.

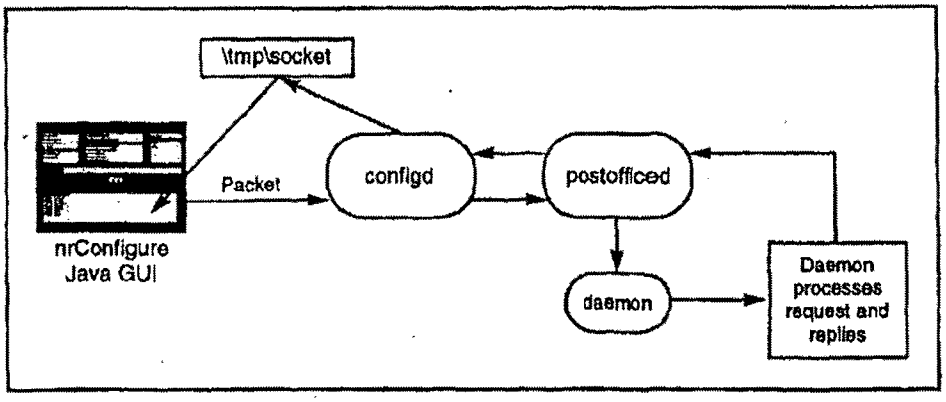


Figure 4.12: nrConfigure Architecture

Starting nrConfigure

nrConfigure is designed to configure daemon applications on an NSX or Director system. This is done by either selecting a machine icon from an OpenView security map or by invoking nrConfigure directly from the command line with the NSX or Director's organization and host id.

- To start nrConfigure from within the Director, choose **Configure** from the Security menu.
- To start nrConfigure from the NSX command line, type the following command:

```
/usr/nr/bin/nrConfigure <Organization ID>.<Host ID>:<Host Name> &
```

NOTE

The `<Organization ID>` and the `<Host ID>` can be found in /usr/nr/etc/hosts. You may run multiple copies of nrConfigure at the same time.

Quitting nrConfigure

To quit any nrConfigure GUI, press the **Cancel** button. Pressing the **Cancel** button on one GUI will not quit any other nrConfigure GUI session.

Help with nrConfigure

If you need help with nrConfigure, follow these steps:

1. **Choose an Application from the nrConfigure GUI.**

2. **Press the Help Button.**

   A help window is displayed for the selected application. Help is also listed for all of the application-supported tokens.

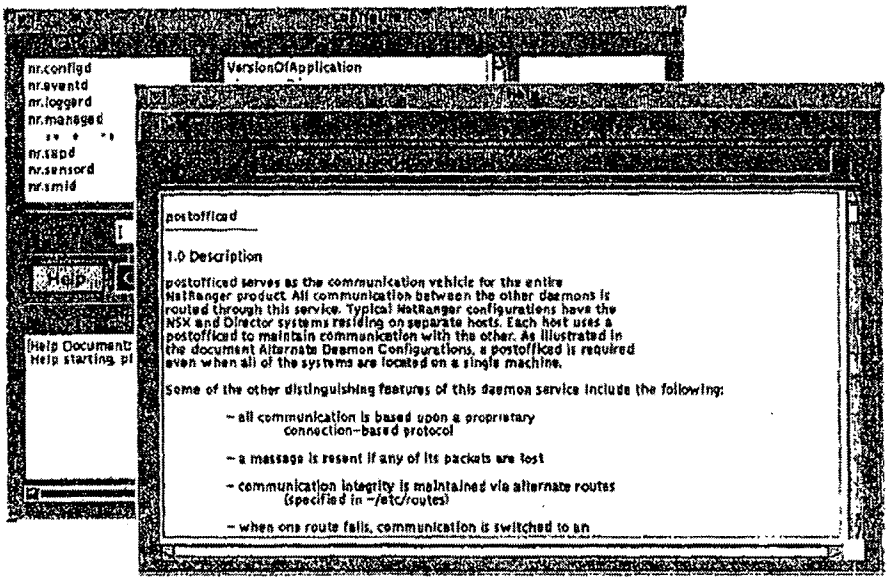


Figure 4.13: nrConfigure Help

Configuring an Application with nrConfigure

To configure an application with nrConfigure, follow these steps:

1. **Select the Application you wish to configure in the Application selection box.**

   The GUI will display that application's allowed Tokens.

2. **Select the Token you wish to execute in the Token Selection box.**

   The GUI will display that token's allowed actions.

3. **Select the Action you wish to execute in the Action selection box.**

The GUI will display the following information:

- Any Optional Parameters, if required
- Any default values for Optional Parameters
- Any set values for Optional Parameters

NOTE

The GUI will display Parameters without current values as the parameter's name in "<>" brackets.

4. **Press the Execute button.**

   The results will display in the Results field.

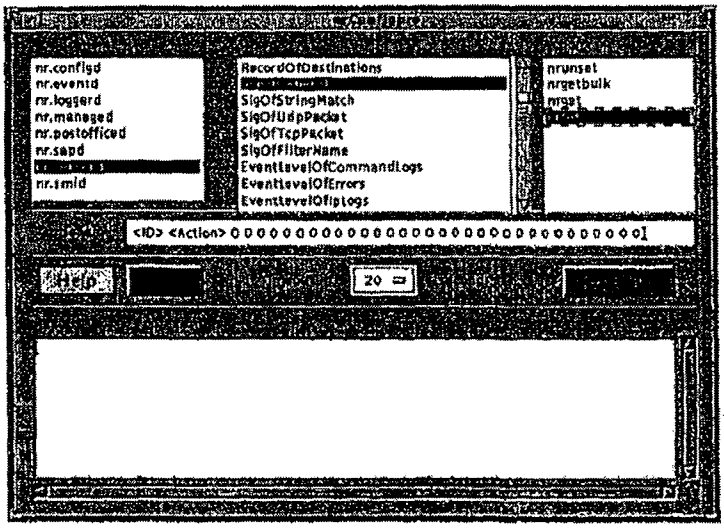


Figure 4.14: Example of Parameters in Optional Field

NetRanger Token Order in the nrConfigure GUI

The display of NetRanger tokens in nrConfigure is dependent upon the order or placement in the list of NetRanger tokens supported by a NetRanger Application or Daemon.

NetRanger Tokens supported by a NetRanger Application are specified in the application persistence file (/usr/nr/etc/persistence.applications). In the following sample, all the NetRanger Tokens are grouped by function (Filename, Read, and Write).

```
APPLICATION=nr.postofficed
ID=10000
HELP=postofficed.txt
TOKEN=FilenameOfError
PARAMETER=filename
DEFAULT=../var/errors.postofficed
TOKEN=FilenameOfConfig
PARAMETER=filename
DEFAULT=../var/postofficed.conf
TOKEN=FilenameOfHosts
TOKEN=FilenameOfServices
TOKEN=FilenameOfAuths
TOKEN=FilenameOfDestinations
TOKEN=ReadFileConfig
TOKEN=ReadFileHosts
TOKEN=ReadFileServices
TOKEN=ReadFileAuths
TOKEN=ReadFileDestinations
TOKEN=WriteFileConfig
END
```

These same NetRanger Tokens could be grouped by the file they operate on (Config, Hosts, Services, Auths, and Destinations), as in the following example:

```
APPLICATION=nr.postofficed
ID=10000
HELP=postofficed.txt
TOKEN=FilenameOfError
PARAMETER=filename
DEFAULT=../var/errors.postofficed
TOKEN=FilenameOfConfig
PARAMETER=filename
DEFAULT=../var/postofficed.conf
TOKEN=ReadFileConfig
TOKEN=WriteFileConfig
TOKEN=FilenameOfHosts
TOKEN=ReadFileHosts
TOKEN=FilenameOfServices
TOKEN=ReadFileServices
TOKEN=FilenameOfAuths
TOKEN=ReadFileAuths
TOKEN=FilenameOfDestinations
TOKEN=ReadFileDestinations
END
```

To edit the location of a NetRanger token in the list of Supported NetRanger tokens for a NetRanger Application, follow these rules:

1. **The APPLICATION, ID, and HELP header (HEADER) must come first to define a valid NetRanger Application.**

```
APPLICATION=nr.postofficed
ID=10000
HELP=postofficed.txt
```

2. **The END tag closes the definition (TRAILER).**

```
END
```

3. **Always move the trailing PARAMETER and DEFAULT tags, if they exist, for the selected token.**

```
TOKEN=FilenameOfError
PARAMETER=filename
DEFAULT=../var/errors.postofficed
```

4. **NetRanger tokens not defined between the HEADER and TRAILER will not show up in the nrConfigure NetRanger token list when the Application is highlighted.**

eventd

Overview

eventd is a general-purpose service that can execute a user-defined action based on alarm events sent to it. These user-defined actions are typically implemented via a shell script, where the various fields of an alarm record are accessible as `$n` command line arguments. As shown in the shell script below, `$1` is the `MessageType`, `$2` is the `RecordID`, `$3` is `GlobalTime`, and so on. Below is an example of the event script in /usr/nr/bin/eventd that parses message fields common to NetRanger message types.

```
# Parse the message fields common to all NetRanger message types
        MessageType=$(1)
        RecordID=$(2)
        GlobalTime=$(3)
        LocalTime=$(4)
        DateStr=$(5)
        TimeStr=$(6)
        ApplID=$(7)
        HostID=$(8)
        OrgID=$(9)
# Parse the message fields based upon the NetRanger message type
        case $(MessageType) in
                2)      ErrorMessage=$(10)
                        MessageCount=$(11)
                        AlertMessage=`printErrorMessage`
                        ;;
                3)      SrcApplID=$(10)
                        SrcHostID=$(11)
                        SrcOrgID=$(12)
                        CommandMessage=$(13)
                        MessageCount=$(14)
                        AlertMessage=`printCommandLogMessage`
                          ;;
                4)      SrcDirection=$(10)
                        DstDirection=$(11)
                        EventLevel=$(12)
                        EventSigID=$(13)
                        EventSubSigID=$(14)
                        ProtocolType=$(15)
                        if [ "$(ProtocolType)" = "TCP/IP" ]; then
                                SrcIpAddr=$(16)
                                DstIpAddr=$(17)
                                DstIpPort=$(19)
                                SourceAddr=$(20)
                                EventMessage=$(21)
                                MessageCount=$(22)
                        else
                                MessageCount=$(16)
                        fi
```

eventd is generally intended to accommodate batch or background processes, such as e-mail notification. Any of the event fields can be used to determine when a user-defined action should occur. For example, a script could be created that generates a separate log of only Level 5 alarms generated by network traffic from a given source IP address.

Using eventd to set up an e-mail notification system

By default, *eventd* is shipped with an e-mail notification service. (Please note that many pager systems can be enabled via e-mail notification.) *eventd* receives copies of alarms from *smid*, arranges them into a readable format, and then sends e-mail messages to users based on information stored in configuration files. An example e-mail message follows:

```
From netrangr@director1.wheelgroup.com Tue Nov  5 12:01:10 1996
From: netrangr@director1.wheelgroup.com
Date: Tue, 5 Nov 1996 12:07:42 -0600
Subject: Alarm OUT->IN Level 2.

Date: 1996/11/05,12:06:00
NSX: [10001.1.100]
Attack: 8000,502 Level: 2
Addrs: 192.216.46.55:80->207.18.164.150:40134
Names: webcrawler.com->lookout.wheelgroup.com
Sig: 8000 "String Match"
Msg: "golf"
```

Basic Setup

eventd is not configured when the package is installed. To set up *eventd*, do the following:

- Set up event script's configuration file
- Set up *eventd* configuration file
- Set up *smid* configuration file
- Set up *eventd* to start

Set Up the Event Script's Configuration File

The event script converts alarms into e-mail messages and sends them to a configurable destination. *eventd* must be configured before it can send alarms. The default configuration file is a template with sample comments only. Unlike the configuration files for NetRanger's daemon services, the configuration files for the event *script* are located in /usr/nr/bin/eventd.

NOTE

The location of the e-mail executables and associated configuration files is totally user-configurable. In this case, these files reside in /usr/nr/bin/eventd. Location and creation of such files is implementation-specific.

Perform the following steps to configure the event script:

1. **Add your organization definition.**

```
ORGANIZATION    100
```

2. **Add users to receive Type 2 alarms. Only one line for Type 2 alarms is accepted.**

```
2       netrangr
```

3. **Add users to receive Type 3 alarms. Only one line for Type 3 alarms is accepted.**

```
3       netrangr
```

4. **Add users to receive Type 4 alarms. Multiple lines may be added for Type 4 alarms. Type 4 alarms have 5 levels, a source, a destination, and recipients.**

```
# Level 4 Alarms
#
# Source and Destination of Alarm
# Src: OUT,IN,-
# Dst: OUT,IN,-
# "-" denotes either
#
#Type   #Level   #Src    #Dst    #Recipients
4       1       OUT     IN      user2
4       2       OUT     IN      user2
4       3       OUT     IN      user2
4       4       OUT     IN      root, user1
4       5       OUT     IN      root, user1
```

- **Level:** The level of the alarm you wish to select. Source and Destination have three possible settings: IN, OUT, and -.
  - **IN.** The address must be inside the network.
  - **OUT.** The address must be from inside the network.
  - **The dash (-) symbol.** The address can be either IN or OUT.
- **Source.** The location of the IP address you wish to select for the recipients

- **Destination.** The location of the IP address you wish to select for the recipients
- **Recipients.** The mail destination of events that match the preceding selection process

Set Up eventd's Configuration File

Before *eventd* can send alarm levels to a destination, the following line in /usr/nr/etc/eventd.conf must be added:

```
EventApplication <UniqueID> <lowestLevelToSend> <locationOfScriptRelativeTo/usr/nr/bin>
```

The following example defines one script for one organization:

```
EventApplication 1002 2 ./eventd/event
```

Set Up smid's Configuration File

In order for *eventd* to receive alarm events from *smid*, the configuration file smid.conf must contain a `DupDestination` entry for *eventd*. An example entry might look as follows:

```
DupDestination director1.wheelgroup eventd  2 ERRORS,COMMANDS,EVENTS
```

Set Up eventd to Start

Edit /usr/nr/etc/daemons by uncommenting the following line:

```
# nr.eventd
```

NOTE

If nr.eventd is left commented or out of /usr/nr/etc/daemons, the daemon will not be started automatically on system start-up.

Advanced Setup

Monitoring Multiple Organizations

If you are monitoring only one (1) organization, use event.conf as your configuration file. Otherwise, build a separate copy of event and event.conf for each organization. To do this, add links pointing to the original event under the name of the organization and make a copy of event.conf under the same name as the new event script. To edit each organization-specific configuration file, follow these steps.

1. **Create a script file with the same name as the configuration file. Use UNIX link command:**

   ```
   %cd /usr/nr/bin/eventd
   %ln -s ./event ./event_wheelgroup
   ```

   The script file, when run, will look for a configuration file by the same name with ".conf" appended. Use event.conf as an example.

2. **Make a copy and edit it according to your local needs:**

   ```
   %cp event.conf event_wheelgroup.conf
   %vi event_wheelgroup.conf
   ```

3. **Edit /usr/nr/etc/eventd.conf by adding organization designations, for example:**

```
EventApplication 1002 2 ./eventd/event_wheelgroup
EventApplication 1003 2 ./eventd/event_organization2
```

In the example above, two scripts for two organizations are being added. The organizations are designated as wheelgroup and organization2.

For example, a new script for wheelgroup is made under the name of event_wheelgroup, and event_wheelgroup.conf, respectively.The lines look like this before configuration:

```
-rwxr-x---   1 netrangr netrangr    6603 Nov 22 17:19 event
-rwxr-x---   1 netrangr netrangr    1021 Nov 22 15:30 event.conf
```

and like this after configuration:

```
-rwxr-x---    1 netrangr netrangr     6603  Nov 22 17:19    event
-rwxr-x---    1 netrangr netrangr     1021  Nov 22 15:30    event.conf
lrwxr-xr-x    1 netrangr netrangr     5     Nov 22 15:31    event_wheelgroup-> event
-rwxr-x---    1 netrangr netrangr     1305  Nov 23 12:47    event_wheelgroup.conf
lrwxr-xr-x    1 netrangr netrangr     5     Nov 22 15:31    event_organization2-> event
-rwxr-x---    1 netrangr netrangr     1305  Nov 23 12:47    event_organization2.conf
```

Changing the Event Error Notification user

Event Error Notification is in the event script. It looks for its own configuration file, and if it cannot find it, or it has an error processing the alarm, it notifies the person(s) named as `MAIL_FAILURE`. To change user to receive e-mail on problems with the configuration file, edit `event`, in the /usr/nr/bin/eventd directory, in the following manner:

```
MAIL_FAILURE="userToReceiveEmailOnFailure@somehost.com,netrangr"
```

For example:

```
MAIL_FAILURE="bob@alarmsRus.com,postmaster@walrus"
```

In the above example, "postmaster@walrus" is the second user added to the `MAIL_FAILURE` line.

Event Signatures

Event signatures consist of a unique signature identifier (SigID) and a sub-signature identifier (SubID). This section includes a list of the event signatures currently defined in NetRanger. SigIDs are built into NetRanger; SubIDs change depending upon the SigID they come under.

New Signatures in the 1.3.1 Release

SIGID	Data Context Provided	Level	Description
1100	No	5	IP Fragment Attack—Detects an attack that exploits a vulnerability in TCP/IP stacks with IP fragments
1101	No	2	Unknown IP Protocol—Alarms on any IP packet with a protocol number that RFC1700 declares reserved or unassigned. This range includes 0, 55-60, and 101-255.
2152	No	4	ICMP Flood—Will fire off if the threshold of X ICMP pkts/sec from a single host is exceeded. The SubSigID will be set to the ICMP type code of the last packet.
3105	Yes	3	Sendmail decode—Detects any mail recipient or sender with the name decode. decode is a mail alias that is misused by many intruders to penetrate systems. It does have a legitimate use for automatically decoding mail messages, but may be abused.
3150	Yes	5	FTP Remote command execution—Detects use of the FTP SITE command. This command has been misused to penetrate systems. It does not normally occur within legitimate FTP sessions.
3151	Yes	3	FTP Reconnaissance—Detects use of the FTP SYST command. It provides information about the server that an intruder may use.
3152	Yes	4	FTP CWD ~root exploit—Detects the exploit used to penetrate systems.
3153	Yes	4	FTP PORT command with address other than the client's. Most PORT commands specify a destination address for the server's data connection to be the client. A different address may be specified, but this is rare. Intruders may use this capability for network recon.
3154	Yes	4	FTP PORT command with a reserved port (under 1024). Legitimate data connections do not usually use a reserved port. Intruders may use this capability for network recon.
The next three signatures alert that a URL that ends with one of these three letter extensions is being activated by a user. Users browsing with MS Internet Explorer are vulnerable to these links.			

SIGID	Data Context Provided	Level	Description
3202	Yes	4	User accesses a .url link
3203	Yes	4	User accesses a .lnk link
3204	Yes	4	User accesses a .bat link
3205	Yes	2	HTML file has .url link (context info provided)
3206	Yes	2	HTML file has .lnk link (context info provided)
3207	Yes	2	HTML file has .bat link (context info provided)
3300	No	5	NetBIOS OOB data attack
3500	Yes	5	Attempt to rlogin into system with user name of '-froot'. Detects an exploit used to penetrate AIX systems.
4050	No	3	UDP Bomb—This attack may cause a denial-of-service by crashing the target system the UDP bomb is going to. The SubSigID is the difference in bytes between the IP data length and the UDP datagram length.
6200	Yes	5	ident buffer overflow—Detects improper ident reply from a server that may overflow a buffer, giving access to an intruder.
6201	Yes	5	ident newline—Similar problem to that mentioned above; gives access.
6202	Yes	5	ident improper request—Improper request to an ident server.
The next four signatures indicate that a user has multiple failed authentication attempts.			
6250	Yes	3	FTP authentication failure (3 failed attempts)
6251	Yes	3	Telnet authentication failure (3 failed attempts)
6252	Yes	3	rlogin authentication failure (3 failed attempts)
6253	Yes	3	POP3 authentication failure (3 failed attempts)

Modified Signatures

SigID	Context-based?	Modification
2151	No	Increased size of allowed ICMP traffic
2000–2012	No	SubSigID now contains ICMP code value

TCP/IP Event Signatures

TCP/IP event signatures are currently divided into three groups: **context-, content-, regular expression-** and **NetSentry-based.**

Context-Based Signatures

Context-based signatures are based on information passed in the TCP/IP header. This can include such things as the destination port, e.g., TCP port 80 for WWW traffic, IP Options, (e.g., source routing), or a combination of events found in a hacking attack. The following group of signatures are context-based attacks that are detectable using NetRanger. Signatures are generated from stateful multi-header packet analysis. NetRanger detects stateful attacks by remembering a sequence of events, or by reconstructing packets to find certain strings or attacks. Those attacks for which alarming is dependent upon maintaining the state of a connection are indicated by a "" following the attack name. These signatures currently analyze the following types of events:

- **Source Routing.** NetRanger detects both loose and strict source routing which is commonly used by hackers to bypass rules found in filtering routers.
- **ICMP Network Sweeps.** This attack uses the ICMP protocol to discover which machines are alive on a remote network. This is most often used as the first step of an attack to find potential targets. This can be implemented three different ways using different ICMP types. All three are detected within NetRanger.
- **Fragmented ICMP traffic.** To get around filtering on large ICMP traffic, it is possible to fragment this traffic and "trick" some intrusion detection systems. NetRanger checks for and alarms on this activity.
- **Large ICMP traffic.** Numerous computers are vulnerable to an attack where if you send an ICMP packet with an extremely large data size it will crash the machine. NetRanger blocks and alarms this traffic.
- **TCP Port Sweep.** When targeting a specific machine, a hacker will frequently run a TCP port sweep to get a list of all available services on the remote target.
- **Half-Open SYN Attack.** This attack was recently publicized when it was used to shut down several Internet Service Providers. This attack can crash a machine by overloading it with TCP connection requests that it never closes.
- **TCP Hijacking.** This attack looks for a characteristic of attacks which take over an existing TCP connection.
- **UDP Port Scan.** When targeting a specific machine, a hacker will frequently run a UDP port sweep to get a list of all available services on the remote target.
- **SATAN Scan.** This looks for both the normal and heavy SATAN attacks.

Content-Based Signatures

For these attacks, NetRanger looks further than simple TCP/IP header information. It actually looks inside the packet for data which indicates an attack in progress. Most of these signatures take advantage of looking for these signatures within a certain context. For example, Sendmail attacks look for certain strings only within the Sendmail port 25.

- **Smail attack.** NetRanger looks for an attack which was only found in the e-mail package "smail".
- **Sendmail invalid recipient.** This looks for attacks which try to send an e-mail to a program on a remote machine. The attack expects the remote machine to execute the e-mail as if it were a program.
- **Sendmail invalid sender.** This attack is like the previous attack except that the sender of the e-mail appears to be a program. When an error occurs within the e-mail the remote machine attempts to return it to the original sender. The e-mail is then executed as a program on the remote machine.
- **Sendmail reconnaissance.** This attack is used to gather information about remote users through the mail port. This is usually used as a preface to other attacks.
- **TFTP password.** This looks for anyone attempting to get the password file using the TFTP service which requires no authentication.
- **DNS HINFO Requests.** This attack uses DNS to gather information about a specific host.
- **DNS Zone Transfer Request.** This attack attempts to gather information about all hosts registered with your DNS server. This can be used by hackers to try to get a map of your network.
- **DNS request for all records.** This attack requests all records maintained on a remote server and it is used to gather information for a future attack.
- **RPC port registration.** This is used to register a specific application to a port on a remote machine and should never occur across the network.
- **RPC port unregistration.** This is used to unregister a specific application to a port on a remote machine and should never occur across the network.
- **RPC dump.** This is used to query a remote machine about which services are running at what ports. This is commonly used by hackers as a preface to an attack.
- **Proxied RPC request.** This is an attack where you trick the remote machine into issuing a request to a service such as NFS for you. This makes the source address appear to be the local machine instead of the attacking machine.

- **NFS *mountd* request.** This is an attack where a remote user tries to connect to the *mountd* process. This attack can determine what file systems a site is sharing on the network and how it might be exploited.
- ***rexd* request.** This is a request to the remote execution daemon and is used to run programs on a remote machine without authentication.
- **YP attacks.** There are five separate attacks which NetRanger looks for which try to take advantage of the service which maintains network passwords and other files.
- **loadmodule attack.** NetRanger looks for a string characteristic of an attack which tricks a set uid program into giving system administrator privileges to the attacker.
- **Any matched string.** NetRanger can look for any string within any service using regular expression matching. This allows any organization to define their own requirements and look for abuse of their proprietary systems.

IP Options Events

IP options consist of a variable-length list of optional information for an IP datagram. Options are rarely used and not all routers and hosts support them. A good security measure is to refuse routing IP datagrams with options. Detecting IP options from externally connected networks provide a good indicator of potential network problems and attacks.

1000 IP options—Bad option list

SubID Values: 0

Misc Field Info: none

Recommended Alarm Value: 1

The list of IP options for the datagram is incomplete or malformed. This may indicate an attack where the attacker is using improperly developed hacking software.

1001 IP options—Record packet route

SubID Values: 0

Misc Field Info: none

Recommended Alarm Value: 2

This option instructs each router to record in the options list the IP address of the interface that the packet will be transmitted from. Because of the limited size of an IP header, only nine addresses can be stored within this list.

1002 IP options—Timestamp

SubID Values: 0

Misc Field Info: none

Recommended Alarm Value: 1

This option is similar to the record route option. Routers are requested to add timestamps into the options list. This option is of little value because of the limited size of the options field.

1003 IP options—Provide s, c, h, tcc

SubID Values: 0

Misc Field Info: none

Recommended Alarm Value: 1

Basic security options as defined in RFC 1038. Used to carry security level and accrediting authority flags. Implementation of security via these options is obsolete and should not be used.

1004 IP options—Loose source route

SubID Values: 0

Misc Field Info: none

Recommended Alarm Value: 5

This option specifies a list of IP addresses that the IP datagram must traverse (excluding the routers the datagram can also pass through). An attacker may transmit datagrams into a network using spoofed source addresses that appear to come from the target network. Responses to these packets are transmitted back to the attacker because the host recognizes the source route option. This allows attackers to defeat IP address based authentication mechanisms. Source route attacks usually use *loose routing* due to the number of hops a datagram must traverse across the Internet.

1005 IP options—SATNET ID

SubID Values: 0

Misc Field Info: none

Recommended Alarm Value: 1

Stream identifier option. This option is obsolete and should not be encountered.

1006 IP options—Strict source route

SubID Values: 0

Misc Field Info: none

Recommended Alarm Value: 5

This option specifies the exact list of IP addresses that the IP datagram must traverse. This is similar to loose source routing. Source route attacks usually do not use this type of source routing.

ICMP Events

ICMP datagrams are generated to provide administrative and diagnostic information. The most common use of ICMP datagrams is the "ping" program, which verifies the existence of a host. ICMP traffic can provide much information about a network, and it can also modify network characteristics such as routing tables. Attackers use ICMP to discover and modify this information. SubIDs for the following signatures are set to zero.

2000 ICMP Echo Reply

SubID Values: 0

Misc Field Info: none

Recommended Alarm Value: 1

This message is generated in response to an echo request. This type of datagram is sent from the host being pinged.

2001 ICMP Unreachable

SubID Values: 0

Misc Field Info: none

Recommended Alarm Value: 1

This message is transmitted to a host stating that the intended destination is unreachable for the IP datagram it transmitted. The first 64 bits of the failed datagram is transmitted along with the unreachable message. Unreachable messages can be used to disrupt existing TCP sessions.

2002 ICMP Source Quench

SubID Values: 0

Misc Field Info: none

Recommended Alarm Value: 2

This message is transmitted to a host to report network congestion and requests a reduction in the current rate of datagram transmission. While not all systems support this feature, it can be used to enact a denial-of-service attack. The performance of the targeted host can be compromised by sending a continuous stream of these source quench messages.

2003 ICMP Redirect (change a route)

SubID Values: 0

Misc Field Info: none

Recommended Alarm Value: 1

This message is transmitted from a router to host with an update for the host's routing table. This enables hosts to start with the minimal routing configuration that is subsequently updated by the network's router tables. Attacker's use redirect messages to place incorrect routes into a target host's routing table to support IP hijacking and other attacks.

2004 ICMP Echo Request

SubID Values: 0

Misc Field Info: none

Recommended Alarm Value: 1

This message requests that the destination host transmit back an echo reply message. This type of datagram is sent to the host being pinged. Individual requests are not a security threat. Large numbers of these requests may be a denial of service attack or network reconnaissance as depicted in SigID 2100 below.

2005 ICMP Time Exceeded for a Datagram

SubID Values: 0

Misc Field Info: none

Recommended Alarm Value: 2

This message was designed to report circular or excessively long routes. A router decrements the time-to-live (TTL) counter whenever it processes a datagram and discards the datagram when the count reaches zero. The first 64 bits of the discarded datagram are transmitted along with the time exceeded message to the originating host. These messages typically occur when an attacker is using the "traceroute" program to help map out a target network. It also occurs when a host has the default TTL set to low (e.g., 30) and transmits datagrams to a destination far across the Internet.

2006 ICMP Parameter Problem on Datagram

SubID Values: 0

Misc Field Info: none

Recommended Alarm Value: 1

This message is transmitted when a datagram header is incorrect. The first 64 bits of the incorrect header is also sent.

2007 ICMP Timestamp Request

SubID Values: 0

Misc Field Info: none

Recommended Alarm Value: 2

This message requests the destination host to transmit back a timestamp reply containing the host's current time. Individual requests are not a security threat. Large numbers of these requests may be a denial of service attack or network reconnaissance as depicted in SigID 2101 below.

2008 ICMP Timestamp Reply

SubID Values: 0

Misc Field Info: none

Recommended Alarm Value: 1

The message generated in response to a timestamp request. Use of this feature could be considered as another type of "ping".

2009 ICMP Information Request (obsolete)

SubID Values: 0

Misc Field Info: none

Recommended Alarm Value: 2

This type of ICMP packet is obsolete and should not be used.

2010 ICMP Information Reply (obsolete)

SubID Values: 0

Misc Field Info: none

Recommended Alarm Value: 1

This type of ICMP packet is obsolete and should not be used.

2011 ICMP Address Mask Request

SubID Values: 0

Misc Field Info: none

Recommended Alarm Value: 2

This message asks the destination host to transmit back the address mask in use on the network. This service was designed to allow diskless clients to set the address mask by broadcasting this request over the local network. Individual requests are not a security threat. Large numbers of these requests may be a denial of service attack or network reconnaissance as depicted in SigID 2102 below.

2012 ICMP Address Mask Reply

SubID Values: 0

Misc Field Info: none

Recommended Alarm Value: 1

The message generated in response to an address mask request. RFC 950 added this feature, but RFC 1122 forbids a host from sending replies unless it has been explicitly configured as an authoritative agent for address masks. Hosts that do not implement this feature correctly may respond to a targeted request—another type of "ping".

2100 ICMP network sweep w/Echo

SubID Values: 0

Misc Field Info: none

Recommended Alarm Value: 5

This signature identifies a host that is transmitting ICMP echo request datagrams to multiple hosts on the network. This method is commonly used by attackers to identify active hosts within a network address range. While this can be a serious attack, network management tools such as HP OpenView also perform this type of network discovery on a regular basis.

2101 ICMP network sweep w/Timestamp

SubID Values: 0

Misc Field Info: none

Recommended Alarm Value: 5

This signature identifies an attacking host that has transmitted ICMP timestamp request datagrams to multiple hosts on the network. While this request can also be used to identify active hosts within a network address range, most attackers use the more common echo request method (see SigID 2100).

2102 ICMP network sweep w/Address Mask

SubID Values: 0

Misc Field Info: none

Recommended Alarm Value: 5

This signature identifies an attacking host that has transmitted ICMP address mask request datagrams to multiple hosts on the network. While this request can also be used to identify active hosts within a network address range, most attackers use the more common echo request method (see SigID 2100).

2150 Fragmented ICMP Traffic

SubID Values: 0

Misc Field Info: none

Recommended Alarm Value: 4

This signature identifies an attacking host that has transmitted a fragmented ICMP packet. By design, ICMP packets are small and should never be fragmented. There are attacks that utilize fragmented ICMP packets to crash target systems.

2151 Large ICMP Traffic

SubID Values: 0

Misc Field Info: none

Recommended Alarm Value: 4

This signature identifies an attacking host that has transmitted a large ICMP packet. By design, ICMP packets are small and should never be fragmented. There are attacks that utilize large ICMP packets to crash target systems.

3000 TCP Connection Logging

SubID Values:0	-	65536	[SYN] Destination TCP port
100000	-	165536	[SYN-ACK] Destination TCP port + 100000
200000	-	265536	[FIN] Destination TCP port + 200000
300000	-	365536	[RST] Destination TCP port + 300000

Misc Field Info: TCP sequence number

Recommended Alarm Value: 1

Packets Required: TCP packets with the SYN, FIN, or RST flags set

This event is used for logging TCP traffic. The token ***LevelOfTrafficLogging*** is used to configure the level of logging.

Level 1: No TCP logging occurs.

Level 2: Only TCP SYN packets are logged (default). This indicates that the source host has initiated an attempt to establish a TCP connection to the destination host using the TCP ports specified. The SubID for this signature is the destination TCP port. If the destination host refuses this connection because the requested port does not exist, an ICMP unreachable message will immediately follow.

Level 3: All TCP SYN, FIN, and RST packets are logged.

The sequence numbers stored in the miscellaneous field provide enough information to determine the number of bytes transferred within a TCP connection. This is accomplished by subtracting the initial sequence number from the SYN packet from the final sequence number from the corresponding FIN packet.

3001 TCP port sweep

SubID Values: 0

Misc Field Info: none

Recommended Alarm Value: 5

This signature identifies an attacking host that has initiated a series of TCP connections to a number of different destination ports on the target host. This method is used by attackers to determine the services available on the target host for potential exploitation.

3050 Half-open SYN attack

SubID Values: 0

Misc Field Info: Affected TCP port on target system

Recommended Alarm Value: 5

This signature identifies the targeted host and TCP port where the source address and port may be randomly generated by an attacker. Detection of this signature is currently limited to FTP, Telnet, WWW, and e-mail servers.

3100 Small attack

SubID Values: 0

Misc Field Info: "to: bounce"

Recommended Alarm Value: 4

This signature detects the very common "smail" attack against e-mail servers.

3101 Sendmail Invalid Recipient

SubID Values: 0

Misc Field Info: "to: |"

Recommended Alarm Value: 4

This signature detects any mail message that is transmitted to an address of "pipe" something. Due to the vulnerabilities previously discovered in sendmail and the complexity of the software, destination addresses of this type should not be allowed.

3102 Sendmail Invalid Sender

SubID Values: 0

Misc Field Info: "from: |"

Recommended Alarm Value: 4

This signature detects any mail message that is transmitted with a return address of "pipe" something. This should not be allowed under any circumstance.

3103 Sendmail Reconnaissance

SubID Values: 0

Misc Field Info: "vrfy" or "expn"

Recommended Alarm Value: 2

This represents a reconnaissance attempt by an intruder. It may also represent an advanced user attempting to determine the mailing address of a friend or co-worker.

3104 Archaic Sendmail Attacks

SubID Values: 0

Misc Field Info: "wiz" or "debug"

Recommended Alarm Value: 2

This sendmail attack is archaic and should not work against current versions of Sendmail.

3200 WWW phf attack

SubID Values: 0

Misc Field Info: none

Recommended Alarm Value: 5

This serious attack is used to exploit the phf program released with the NSCA and Apache web servers.

3201 WWW General cgi-bin attack

SubID Values: 0

Misc Field Info: none

Recommended Alarm Value: 5

This signature detects any cgi-bin script that attempts to retrieve the file /etc/passwd.

3250 TCP Hijacking

SubID Values: 0

Misc Field Info: none

Recommended Alarm Value: 5

This signature analyzes both streams of data within a TCP connection to detect when TCP hijacking may have occurred. This current implementation of this signature does not detect all types of TCP hijacking and false positives may occur. Even when hijacking is discovered, little information is available to the operator other than the source and destination addresses and ports of the systems being affected.

4000 UDP packet

SubID Values: 0-65536 Destination UDP port

Misc Field Info: none

Recommended Alarm Value: 1

This event is used for logging UDP traffic. The token ***LevelOfTrafficLogging*** is used to configure the level of logging.

Level 1: No UDP logging occurs.

Level 2: This message records that the source host has sent a UDP datagram to the destination host using the UDP ports specified. The SubID for this signature is the destination UDP port. If the destination host does not have a UDP service at the requested port, an ICMP unreachable message will immediately follow. The default configuration is to not generate this signature unless it is explicitly specified within the configuration file.

Level 3: Same as level 2.

4001 UDP port sweep

SubID Values: 0

Misc Field Info: none

Recommended Alarm Value: 5

This could be a serious attack. This signature identifies an attacking host that has sent several UDP datagrams to a number of different destination ports on the target host. This method is used by attackers to determine the services available on the target host for potential exploitation.

4100 TFTP Passwd File

SubID Values: 0

Misc Field Info: none

Recommended Alarm Value: 5

Packets Required: A large sampling of UDP packets

This signature detects an attempt to access the passwd file via TFTP

6001 Normal SATAN probe

SubID Values: 0

Misc Field Info: none

Recommended Alarm Value: 5

This signature is generated when an attacking host has run the tool "SATAN" in normal mode against a target host on the network. Other types of attack activity similar to this method may also cause this signature to be generated.

6002 Heavy SATAN probe

SubID Values: 0

Misc Field Info: none

Recommended Alarm Value: 5

This signature is generated when an attacking host has run the tool "SATAN" in heavy mode against a target host on the network. Other types of attack activity similar to this method may also cause this signature to be generated.

6050 DNS HINFO Request

SubID Values: 0

Misc Field Info: DNS query name

Recommended Alarm Value: 2

This signature detects an attempt to access HINFO records from a DNS server. This is commonly used by intruders to determine the types of systems on a network.

6051 DNS Zone Transfer

SubID Values: 0

Misc Field Info: DNS query name

Recommended Alarm Value: 1

This signature detects legitimate DNS zone transfers.

6052 DNS Zone Transfer from High Port

SubID Values: 0

Misc Field Info: DNS query name

Recommended Alarm Value: 4

This signature detects a DNS zone transfer with the source port not equal to 53. This is a common network reconnaissance technique used by intruders.

6053 DNS Request for all Records

SubID Values: 0

Misc Field Info: DNS query name

Recommended Alarm Value: 2

This signature detects a DNS request for all records.

6100 RPC Port Registration

SubID Values: RPC program number

Misc Field Info: none

Recommended Alarm Value: 5

This attack attempts to register new RPC services on a target host.

6101 RPC Port Unregistration

SubID Values: RPC program number

Misc Field Info: none

Recommended Alarm Value: 5

This attack attempts to unregister RPC services on a target host.

6102 RPC Dump

SubID Values: 0

Misc Field Info: none

Recommended Alarm Value: 4

This is an example of network reconnaissance. This is produced by executing a "rpcinfo -p" against a target host.

6103 Proxied RPC Request

SubID Values: RPC program number

Misc Field Info: none

Recommended Alarm Value: 5

This attack exploits a vulnerability of the portmapper by causing it to forward RPC requests to the local system. This may defeat existing authentication mechanisms. The most widespread use of this vulnerability occurs in tricking *mountd* on NFS servers to disclose filehandles.

6150 ypserv Attempt

6151 ypbind Attempt

6152 yppasswdd Attempt

6153 ypupdated Attempt

6154 ypxfrd Attempt

6155 mountd Attempt

6175 rexd Attempt

SubID Values: 0

Misc Field Info: none

Recommended Alarm Value: 5

Signatures 6150 through 6175 detect attempts to access the services specified. It is recommended that these services should not be accessed via the Internet.

Regular Expression-Based Signatures

Regular expression- (regex) based signatures are generated from stateful multi-layer datagram contents analysis. In simple terms, this means arranging the contents of packets and doing string matching on the results. Any regular expression up to 64 characters in length is supported, which provides an unlimited number of signatures.

While all regex-based signatures have a **SigID of 8000,** each string must have a unique SubID assigned to it. Numbering SubIDs is left to the individual installing and maintaining NetRanger. Each string entry also specifies the port number and direction of traffic where it should be searched for. For example, in order to detect attackers using the command "help" on mailservers, the regular expression "[Hh][Ee][Ll][Pp]" should be reported only when it is in the datastream going to port 25 on a target host. This drastically lowers the number of false positives.

Performance is the only limitation to the number of strings that can be simultaneously searched for. Analyzing strings directed at the destination port provides substantially better performance than the reverse. For example, malicious activity inside of TELNET sessions can be determined by what an attacker is typing, and not from what appears on the screen. A typical Telnet session has over 95% of the network traffic generated going from the port 23 to the user's screen. In this example, better performance is achieved by only analyzing the remaining 5%.

Following are examples of configurable strings.

NOTE

In the following tables, the numbers in the **Direction** column indicate the following: 1=To Port, 2=From Port, 3=Both. The numbers in the **Occurrences** column indicate how many times the string match occurs before an alarm is sent.

The following string detects users attempting to FTP the password file.

FTP	SIGID	Port	Direction	Occurrences	String Match
RecordOfStringName	2101	21	1	1	"[Rr][Ee][Tt][Rr][ ]+passwd"

The following strings are used to detect the username and passwords of users in ftp sessions. These also indicate when directories are created/deleted and when files are GET/PUT. These are not security relevant and are provided only to show installers examples of what can be monitored. Capture of user passwords may be a security risk and WheelGroup recommends against it (i.e., what happens if the log file containing cleartext passwords was world readable on a system, printed out, or lost).

FTP	SIGID	Port	Direction	Occurrences	String Match
RecordOfStringName	2102	21	1	1	"[Mm][Kk][Dd]"
RecordOfStringName	2103	21	1	1	"[Rr][Mm][Dd]"
RecordOfStringName	2104	21	1	1	"[Uu][Ss][Ee][Rr]"
RecordOfStringName	2105	21	1	1	"[Pp][Aa][Ss][Ss]"
RecordOfStringName	2106	21	1	1	"[Rr][Ee][Tt][Rr]"
RecordOfStringName	2107	21	1	1	"[Ss][Tt][Oo][Rr]"

The following strings are used to detect loadmodule/eqvload attacks and exploitation of the Berkeley R-commands. These are representative strings of host-based attacks. Additional strings may be added to detect other host-based malicious activity.

Telnet	SIGID	Port	Direction	Occurrences	String Match
RecordOfStringName	2301	23	3	1	"IFS[=]+[/]"
RecordOfStringName	2302	23	1	1	"[/]etc[/]shadow"
RecordOfStringName	2303	23	1	1	"[+][ ]+[+]"

The following example shows how to log every HTTP GET on a network. The context info provided with the string will show what URL was used in the GET.

Web	SIGID	Port	Direction	Occurrences	String Match
RecordOfStringName	8001	80	1	1	"GET[ ]+"

The following example shows how to detect if users are running Microsoft Internet Explorer™. If an organization has a policy of only using Netscape™, detection of MSIE could be enabled generate a security violation.

Web	SIGID	Port	Directions	Occurrences	String Match
RecordOfStringName	8002	80	1	1	"User-Agent."MSIE"

The following strings are used to detect usernames/passwords of POP3 mail users. As mentioned above, these are not very security relevant.

POP3	SIGID	Port	Direction	Occurrences	String Match
RecordOfStringName	11001	110	1	1	"[Uu][Ss][Ee][Rr]"
RecordOfStringName	11002	110	1	1	"[Pp][Aa][Ss][Ss]"

The following string is used to detect the name of a user associated with the connection queried in the Ident request.

Ident	SIGID	Port	Direction	Occurrences	String Match
RecordOfStringName	11301	113	2	1	"USER"

The following examples represent another way to detect host based attacks. Use the same strings as in a Telnet session.

rlogin	SIGID	Port	Direction	Occurrences	String Match
RecordOfStringName	51301	513	3	1	"IFS[=]+[/]"
RecordOfStringName	51302	513	1	1	"[/]etc[/]shadow"
RecordOfStringName	51303	513	1	1	"[+][+]"

NetSentry-Based Signatures

NetSentry-based signatures are generated by reporting packet failures and/or successes from NetSentry filters. When the NSG BorderGuard is used to enforce a specific security policy, an option exists to copy a subset of failed packets from the BorderGuard to the NSX. Each copied packet includes the name of the NetSentry filter that failed the packet. NetRanger matches this name with a pre-defined list of filter names and generates a security violation event record upon a match.

```
filter incom1_ip_spoof_fail
ip_sa mask 0xFF000000 in (10.*.*.*)
copy_to 10.1.5.3 35399
fail;
end
```

The above filter was designed to analyze all packets entering a network. If the source address is from the network, it is an impossible packet and may be part of an IP spoofing attack. The filter will send a copy to the NetRanger/NSX at address 10.1.5.3, and then fail the packet. NetRanger will match the string "incom1_ip_spoof_fail" and generate a security violation event record. The event record will include the source and destination IP addresses and applicable ports.

Using this NetRanger capability, any filter on the BorderGuard is capable of generating event signatures. Because the BorderGuard allows a large number of highly configurable filters using the NetSentry filter language, the number of potential NetSentry event signatures generated by NetRanger could be considered unlimited. NetSentry filternames for versions 2.0 thru 3.1 of the BorderGuard OS is case sensitive. Version 4 of the BorderGuard OS and the ERS report filternames in uppercase. If the original filtername is "finger_fail", then NetRanger must be configured to look for the filtername "FILTER_FAIL".

All NetSentry-based event records have a SigID of 10000. Each defined filtername in NetRanger must have a unique SubID assigned to it. Numbering SubIDs is left to the individual installing and maintaining NetRanger. The example security filters that come with NetRanger implement the following:

```
incom1_ip_spoof_failIP spoofing attacks
incom1_ip_source_route_failIP source routing attacks
incom1_tcp_frag_header_failFragmented TCP header attacks
incom1_tcp_small_frag_failUndersized TCP header attacks
incom1_tcp_failGeneric failed TCP connections
incom1_udp_failGeneric failed UDP packets
incom1_ftp_failFailed FTP attempt
incom1_telnet_failFailed TELNET attempt
incom1_smtp_failFailed e-mail attempt
incom1_dns_failFailed Domain Name Server attempt
incom1_gopher_failFailed gopher attempt
incom1_finger_failFailed finger attempt
incom1_www_failFailed WWW attempt
incom1_pop2_failFailed POP2 mail attempt
incom1_pop3_failFailed POP3 mail attempt
incom1_rpc_failFailed RPC attempt
incom1_auth_failFailed identd attempt
incom1_nntp_failFailed network news attempt
incom1_ntp_failFailed network time attempt
incom1_exec_failFailed rexec attempt
incom1_login_failFailed rlogin attempt
incom1_cmd_failFailed rsh attempt
incom1_printer_failFailed printer attempt
incom1_ntalk_failFailed network talk attempt
incom1_uucp_failFailed UUCP attempt
incom1_x11_failFailed X11 attempt
incom1_udp_dns_failFailed DNS packet
incom1_tftp_failFailed TFTP packet
incom1_udp_rpc_failFailed RPC packet
incom1_snmp_failFailed snmp packet
incom1_syslog_failFailed syslog packet
```

THE SECURITY ANALYSIS PACKAGE 5

Collection, Management, and Analysis of NetRanger Data

This chapter presents the following information on the NetRanger Security Analysis Package (SAP):

- *High Level Overview*
- *Functional Overview*
- *Installation*
- *SAP Tutorial*
- *SAP Reference*
- *File Management Tokens*

High Level Overview

The NetRanger Security Analysis Package (SAP) provides data collection, management, and analysis services on both NSX and Director systems. These services represent a key operational component of NetRanger. Figure 5.1 illustrates the relationship among the parts.

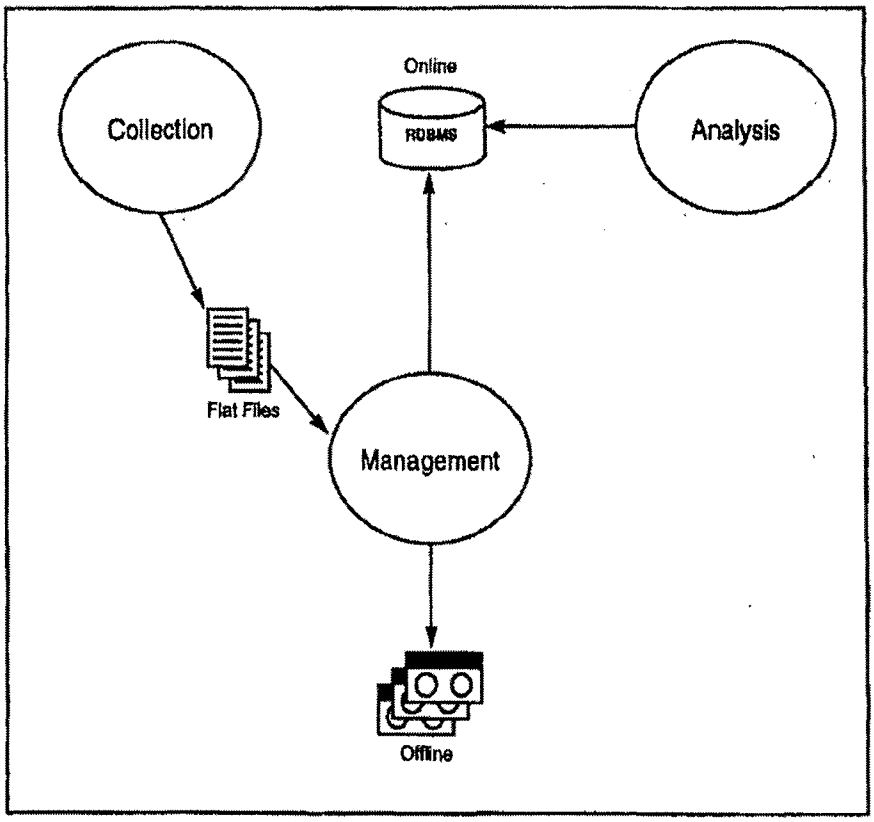


Figure 5.1: Overview of SAP Functions

Collection

SAP collects two basic forms of data: high-level event data and detailed binary IP session data. SAP uses the two types of log files to capture these types of data:

- **Event logs,** which capture *NSX alarms, system commands,* and *system errors.* These are ASCII files written to the /usr/nr/var directory with the naming convention of log.<YYYYMMDDHHMM>. By default, level 2-5 events are written to event logs on a Director system, and low-level 1 events are written to an event log on the NSX system.

- **IP Session logs,** which capture all of the incoming and outgoing TCP packets associated with a specific connection. By definition, these logs contain binary data. They are written to an NSX Sensor's /usr/nr/var/iplog directory with the naming convention of iplog.<src IP address>.

By default, level 1 Event and IP Session logs are left on an NSX Sensor until they are required. This prevents the relatively large amounts of data associated with these types of events from impeding NetRanger communications during periods of network load.

Even though the SAP is designed to transfer NetRanger data into industrial strength databases, both types of data are initially written to flat files for two reasons: ***speed*** and ***fault tolerance.*** Data can be written to a flat file much faster than to a database, and access to flat files is guaranteed as long as the host system is operational. Databases, on the other hand, are subject to unpredictable load, and contain too many access layers to be truly fault tolerant—if the database is down, you've got nowhere to go.

Management

The three main goals of SAP data management are to automatically

- collect data in a fast, fault-tolerant manner;
- stage that data onto database management and archival systems; and
- prevent disk systems from filling up.

These services are easily configured and monitored via **nrConfigure.**

Analysis

Once data is loaded into a database, it can be analyzed for patterns and trends. Status reports relating to network activity and vulnerabilities can also be generated. Although any number of different third-party tools can generate these types of output, SAP is shipped with a small but comprehensive collection of Oracle SQL*Plus queries that show how event data can be analyzed relative to different perspectives, such as time versus events. These queries are described in *SAP Tutorial.*

Note that although Collection and Management are provided by default on an NSX Sensor, only Collection services are provided by default on Director systems. SAP is bundled in this manner for two reasons:

1. The integrity of the file system on an NSX Sensor is a core operational requirement, and the SAP Collection and Management services fulfill this requirement.

2. The Management and Analysis services on a Director system are designed to work with large volumes of data stored in a Relational Database Management System (RDBMS). These services are bundled as a separate component to help keep costs down for those users who do not require this level of functionality.

Table 5.1: Availability of basic SAP services

	Collection	Management	Analysis
NSX	standard	standard: file only	not applicable
Director	standard	optional: file & data	optional

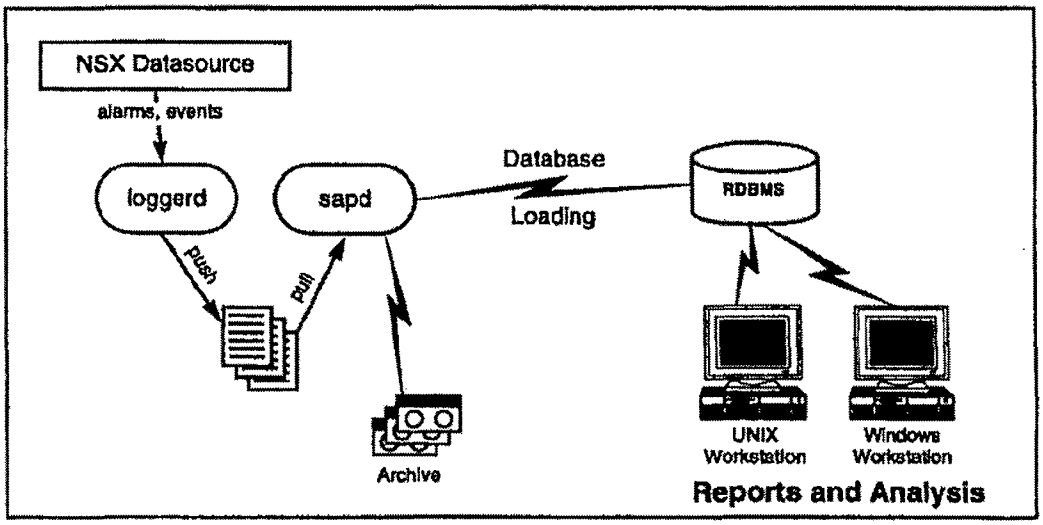


Figure 5.2: Data Collection, Management, and Analysis

Functional Overview

NetRanger uses a simple **push-pull** process to migrate data from flat files to a database or data archive. Figure 5.3 diagrams the push-pull process for both types of log files.

The *push* process uses the *loggerd* service to write *event*, *command*, and *error* notifications into a single flat file in /usr/nr/var. This file is *serialized* based on configurable size and time thresholds set via **nrConfigure**. Once the current log file exceeds one of these thresholds, it is replaced by a fresh log file and moved to /usr/nr/var/new.

The *pull* process is driven by *sapd* service, which relies on its own size and time thresholds. Note that multiple log files accumulate in /usr/nr/var/new whenever *loggerd* serializes log files at a faster rate then *sapd* writes to a database. Fortunately, high data rates tend to be episodic, and the SAP is designed to recover from such loads by processing all of the files in /usr/nr/var/new until the directory is empty. *sapd* pulls the oldest file from /usr/nr/var/new into /usr/nr/var/tmp and executes database load procedures specified by the user. After the log file is successfully loaded into a database, it is placed in /usr/nr/var/dump where it is processed by a user-defined purge process.

This basic push-pull process applies to IP session logs as well as event logs. The primary differences are that IP session logs are written to /usr/nr/var/ipIog rather than /usr/nr/var/new, and they are automatically placed in /usr/nr/var/dump after a configurable amount of time.

A number of different data management scenarios are presented in the *SAP Tutorial* section.

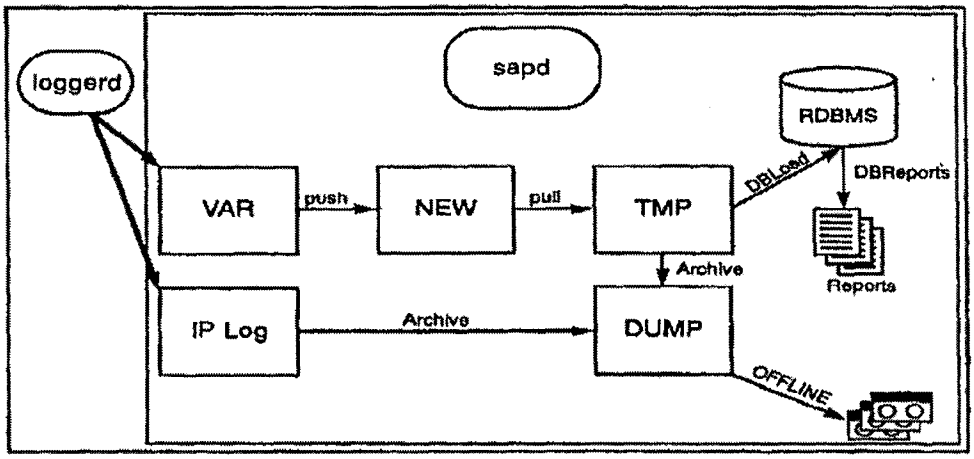


Figure 5.3: The Push-Pull Process

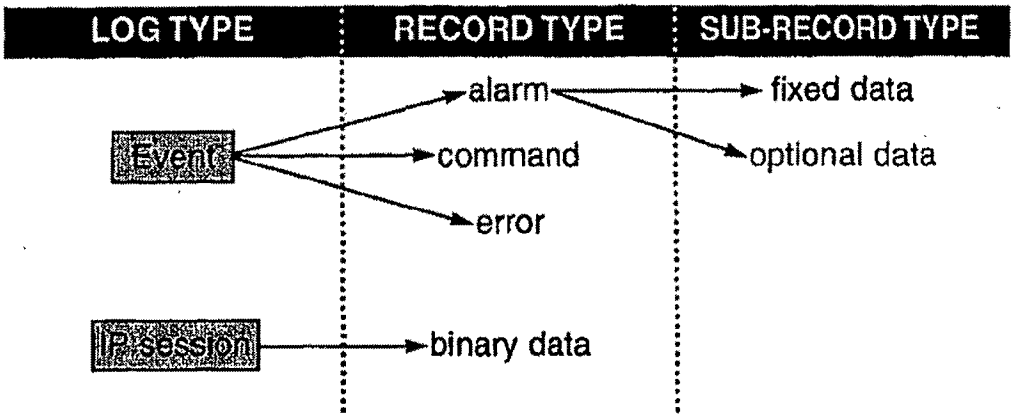


Figure 5.4: Basic Log File Data Relationships

Collection

Event Alarm Data

As noted in the introduction, SAP generates two basic types of log data: Event and IP Session data. Event alarm data is unique in that its record format contains *optional* as well as a *fixed* components.

Fixed Alarm Data

Fixed alarm data includes the following information:

- Which sensor detected the event.
- When the alarm was generated.
- The type of alarm.
- The source and target of the event.

The exact record format for the fixed portion of alarm data is defined in Table 5.5 of the *SAP Reference* at the end of this chapter.

Optional Alarm Data

The *optional* data field contains *ad hoc* information that cannot be described by the fixed portion of the alarm record format. Optional data is *ad hoc* because it depends on the type of alarm record it is associated with. Many context-based alarms, such as port sweeps and ICMP requests, currently do not use this part of the alarm record. Others, such as Security Violation and DNS Request alarms, use this portion of the alarm record to record such information as the name of the policy filter or the request string associated with the event.

A third type of optional data is *Context* data. Not to be confused with context-based alarms, this optional data contains a snapshot of the incoming and outgoing network traffic that leads up to the generation of any type of alarm that includes the exchange of textual data. This data can then be used to reconstruct the events that led up to the unauthorized event.

This type of data cross-cuts the context-content alarm signature classifications defined in Chapter 4. It is generated for such context-based alarms as Telnet and HTTP-Server connection requests as well as such content-based signatures as Sendmail and User-Defined signatures with a SIGID of 8000.

A maximum of 256 bytes of incoming and 256 bytes of outgoing network traffic is associated with an alarm record. The Director's **Security->Show->Context Data** menu option allows you to view the context data for certain alarm levels. The data can also be automatically loaded into the ***nr_log_context*** table at the same time the fixed data is loaded into ***nr_log_alarm.***

As one might expect, there is a performance penalty associated with the logging of context data. SAP provides two configuration parameters to help control this overhead:

MinContextLevel is a *loggerd* token that allows you to specify the minimal alarm level for logging optional context data. For example, context data associated with level 1-3 alarms will not be written to a Director's Event log file when this token is set to "4." Note that this token does not affect access to this information from the Director console. Users can still view this information for alarms that fall below the MinContextLevel threshold.

SAP_EXCLUDE_CONTEXT. The *sapd* process relies on a script named **load_run.sh** to determine exactly what levels and types of event data should be exported onto a database. Context data is not loaded into the ***nr_log_context*** table when this variable is set.

Log File Serialization

As noted earlier, serialization of log files is based on time, size, or combined time/size thresholds specified via **nrConfigure.** The *time* threshold is specified via *loggerd's* **NumberOfSwitchMinutes** token; *size* is specified by the **NumberOfSwitchBytes** token. Appropriate threshold values depend on such factors as network activity, and the amount of disk space allocated for log files.

In the following example, all but one file was serialized on the basis of a time threshold of **one hour**. *log.199611070819* was serialized ahead of schedule because it exceeded the 100-kilobyte *size* threshold before the next hour had passed.

Filesize	Date	Time	Filename
61058	Nov 7	06:19	log.199611070519
60524	Nov 7	07:19	log.199611070619
70008	Nov 7	08:19	log.199611070719
103730	Nov 7	08:40	log.199611070819
76926	Nov 7	09:40	log.199611070840
83476	Nov 7	10:40	log.199611070940

Fail-Safe Features

One of the primary requirements of *loggerd* is that it perform in a fail-safe manner under heavy load. These fail-safe measures currently include safeguards against runaway serialization and loss of files due to overwriting. Users are also able to perform manual serialization.

Runaway Serialization

In order to prevent *loggerd* from creating a large number of log files under heavy load, and possibly exceeding the file management limits of the operating system, *loggerd* makes sure that the current log file remains open for at least one minute. In the following example, log files of 259K and 177K were created between 10:14 and 10:16 because the one-minute threshold had to be met in spite of the size threshold.

Filesize	Date	Time	Filename
100358	Nov 11	09:49	log.199611110916
101432	Nov 11	10:14	log.199611110949
259555	Nov 11	10:15	log.199611111014
177152	Nov 11	10:16	log.199611111015
100307	Nov 11	10:44	log.199611111016

Loss of Files from Overwriting

In addition to preventing runaway serialization, *loggerd* is designed to prevent files from being overwritten when NetRanger services are *started and stopped more than once a minute*. Although uncommon, this type of activity can occur during system maintenance. In this situation *loggerd* appends a sequential identifier to each log file name.

In the following example, the NetRanger daemons were started and stopped four times within a single minute (16:44). As a result, *loggerd* appended the numbers 1, 2, and 3 to the end of the last three files.

```
Filesize    Date     Time     Filename
  178       Nov 8    16:44    log.199611081644
  313       Nov 8    16:44    log.199611081644.1 <--
    0       Nov 8    16:44    log.199611081644.2 <--
  306       Nov 8    16:44    log.199611081644.3 <--
```

Manual Serialization

The current event log file can also be serialized on demand by issuing an **nrexec** command against *loggerd's* **SwitchFile** token. Size and time thresholds are reset at that point.

Management

sapd

sapd is the NetRanger daemon that launches database load or archive processes based on user-defined **conditions** and **actions**. Essentially, an action is launched when its associated condition is satisfied. Condition-action pairs are known as ***triggers***.

Multiple ***triggers*** can be defined via **nrConfigure**, which are evaluated by *sapd* in the order in which they are defined. This supports prioritized processing of triggers. For example, the most important criterion for processing log files might be an upper limit on the amount of disk space that log files are allowed to consume. The next criterion might be a fixed time interval.

Conditions

Much like *loggerd*, *sapd* conditions are based on filesystem and time thresholds.

Filesystem conditions can be based on any of the following thresholds:

- **Absolute number** of log files in the source directory.
- **Cumulative size** of all of the log files in the source directory.
- **Overall percentage of the disk file partition consumed.**

Time conditions are based on either of the following:

- An **elapsed time** interval.
- A **specific day or time** of day.

Actions

The actions that can be launched by *sapd* include

- **DBLOAD**, which loads the contents of an ASCII file into a database management system, such as Oracle or Remedy ARS.
- **ARCHIVE**, which loads a log file into a user-defined destination, such as tape storage. This action includes a built-in *purge* facility that removes the oldest file in /usr/nr/var/dump when a user-defined disk utilization threshold is reached. A bulk-purge facility can also be configured to move a set of log files instead of one at a time. Note that *sapd* will not evaluate another condition until it has completed the current ARCHIVE action.
- **BATCH**, which is a version of ARCHIVE that operates in the background and allows *sapd* to continue evaluating other conditions without *blocking*. This action is intended primarily for launching database reports, which can execute in parallel with other actions. ARCHIVE is better suited for controlling archiving actions, which generally cannot proceed in parallel unless the NSX or Director has access to multiple archival devices.
- **NOTIFY**, which repeatedly launches an action based on a *NotifyInterval* so long as the launch condition holds true. Although of general-purpose construction, this action type was designed to generate e-mail messages when the NSX or Director file system exceeds user-defined thresholds.

sapx

sapx is the default database load program shipped with the SAP package. It currently supports Oracle and the Remedy ARS system. Event log data can be exported to other database management systems, such as Informix by using their native bulk data load utilities. The rules files for Oracle's SQL*Loader utility are included in /usr/nr/bin/sapd/skel for reference.

sapx requires the following parameters in order to load data into a database:

- **LogFileName,** which identifies the directory location and name of the log file.
- **SourceType,** which identifies the event record types to pass onto the target database. Event records are uniquely identified by a numeric marker that is either 2 (error), 3 (system command), or 4 (alarm). *sapx* also supports the record type "5". This is a reserved record identifier for alarm signatures, which are only loaded during initialization or manual update of a database. Also note that although *sapx* can load NetRanger error, command, alarm, and signature records into Oracle, it only loads alarm records into Remedy ARS systems.
- **TargetType,** which identifies the type of database to load the log file in to. The value for Oracle is "1". The value for Remedy ARS is "3".

The *SAP Reference* section provides more information on *sapx*.

Analysis

sapr

As noted in the *High Level Overview* section, SAP is shipped with a small collection of SQL queries that generate simple columnar reports. These reports are referred to as the SAP Reports, or *sapr*, and demonstrate how NetRanger intrusion detection data can be viewed from three basic perspectives: **Space, Time,** and **Events.**

Each of the *sapr* perspectives is effectively a *data dimension* that contains subordinate elements, or *sub-dimensions.* For example, *Time* can be broken down into years, months, days, and so on. *Space* starts with IP addresses and ends with specific ports on a host. Imposing these type of hierarchical structures on NetRanger's event data makes it easier to move from high to low levels of detail as well as shift from one perspective to another. Changing the level of detail is commonly known as **data drilling,** while changing perspective is referred to as **data pivoting.**

Both the dimensions and the associated *sapr* queries are presented in the *SAP Reference* section.

Installation

This section summarizes the steps required to deploy an operational SAP on both the **NSX** and **Director.**

NOTE

HP-UX systems require one change in NetRanger's .profile. Please refer to Appendix D: DBMS REQUIREMENTS AND SETUP for information on changing the stty setting for the @ character.

Before You Install the Security Analysis Package

Collect the following information before you install and configure this version of the SAP:

- Logging Profile
- Serialization Thresholds
- File Management Thresholds
- Target Archive and Dump Destinations
- Notification Intervals, Conditions, and Destinations

Logging Profile

As explained at the beginning of this chapter, there are two basic types of data that can be logged by an NSX sensor: *Events* and *Binary IP Session data.* Although enabling IP Session logs is a simple yes/no proposition, Event logs require a data resolution rule. For example, a possible logging profile may include one of the following:

- Alarms must be level 2 and above
- Information must be level 1 and above
- Full Information must be level 1 with *sensord's* **LevelOfLogging** token set to 3

Serialization Thresholds

As noted previously, *loggerd* serialization thresholds can be time-, size-, or time/size-based. The minimum *time* threshold is one minute. The minimum log *size* threshold is one kilobyte. Some possible settings include:

- **Every minute.** This setting should be used when you want event log data processed as fast as possible, or to store event data in smaller files.
- **Every hour.** This assures that *loggerd* serializes the current log file on an hourly schedule.
- **Every 10 minutes or when the log file exceeds one megabyte in size.**

File Management Thresholds

In order to prevent an NSX Sensor's file system from locking up, file management thresholds must be defined for the following SAP holding areas under /usr/nr/var: **new, iplog,** and **dump.** As with *loggerd,* these thresholds can be specified based on time, size, or a combination of both. The options consist of the following:

- **DirFiles,** which designates a **maximum number of files** in the holding directory (e.g., maximum of 5 files).
- **DirSize,** which designates a **maximum number of kilobytes** consumed in the holding directory (e.g., 2000 KB = 2 MB).
- **DirUsed,** which designates a **maximum percentage of the disk partition** consumed by the holding directory (e.g., 30%). Another way of thinking about this threshold is percentage of the disk partition that is *unused.*
- **CronTime,** which designates a **time of Day** (e.g., every day at 1AM).
- **IdleTime,** which designates an **elapsed Time Interval** (e.g., every 30 minutes).

Each NSX Sensor is shipped with default thresholds, which can be found in /usr/nr/etc/sapd.conf.

Target Archive and Dump Destinations

Although data can be archived and dumped to a storage device (such as a DAT tape), the *default archive process* compresses log files and then moves them to /usr/nr/var/dump. The *default dump process* deletes the oldest file.

- **Archive:** directory or device
- **Dump:** directory, device, or purge process

Notification Interval, Conditions, and Destination

As stated earlier, the SAP can generate e-mail notifications on a regular interval so long as a condition, such as violation of a file system threshold, is true. For example, you may wish to be notified when your NSX /usr/nr/var filesystem exceeds 75% of the disk drive's capacity, or when there is more than 250 megabytes of data in the /usr/nr/var/dump directory. The following information must be defined in order for this facility to work properly:

- **E-mail or pager/e-mail address** (admin@command.com).
- **Conditions to initiate the notification** (see the serialization thresholds listed above).
- **The notification interval,** which is how often to send a notification while the condition is true (e.g., every 20 minutes).

Installing and Configuring SAP on an NSX

The nrConfig utility guides you through a series of menus that assist with installation and configuration of the SAP package.

The first menu in nrConfig is the Feature Selection Menu, and looks something like the following:

```
FEATURE SELECTION MENU
Choose what features you want ENABLED on this host.
(Choosing an 'ENABLED' feature will disable it.)
        1 - NSX
        2 - Director
        3 - Logging
        4 - Database Reporting
        5 - File Management
        6 - Event Paging
        7 - Postoffice Routing
        8 - Configuration Control
    Enter - CONTINUE
Feature # >
```

From this menu, you can enable and disable various features. When you select a feature from the menu, it is prefixed with ENABLED to show that it has been selected. To disable a selected feature, simply choose it from the menu again.

To install and configure SAP on an NSX sensor, ensure that Item 2 in the Feature Selection Menu (Director) is **not** enabled. Enable logging, database reporting, and file management features. The menu will look something like the following:

```
FEATURE SELECTION MENU
Choose what features you want ENABLED on this host.
(Choosing an 'ENABLED' feature will disable it.)
ENABLED 1 - NSX
        2 - Director
ENABLED 3 - Logging
        4 - Database Reporting
ENABLED 5 - File Management
        6 - Event Paging
ENABLED 7 - Postoffice Routing
        8 - Configuration Control
    Enter - CONTINUE
Feature # >
```

Next, press Enter to return to the Main Menu, which will look like the following:

```
MAIN MENU                    NetRanger Configuration Version 1.3.1
Choose what Section you want to configure.
          1 - Select Features
          2 - Host Address Information
          3 - Sensor Configuration
          4 - N/A (Database Configuration)
          5 - Source Configuration
          6 - Destination Configuration
          7 - Postoffice Router Configuration
          8 - N/A (Sleeve Configuration)
          9 - Clear Temporary Configuration Files
         10 - Generate Temporary Configuration Files
         11 - Edit/Review Temporary Configuration Files
         12 - Review Temporary Configuration Files
         13 - Commit Temporary Configuration Files
      Enter - EXIT
      Section # >
```

Press 4 for Database Configuration, which will guide you through the process of entering local *sapd* token values. The menu for Database Configuration looks like the following:

```
 Database USER ID :
Database PASSWORD :
    Notify Person :
DATABASE CONFIGURATION MENU
        1 - Database USER ID
        2 - Database PASSWORD
        3 - Notify Person
    Enter - CONTINUE
Select Database Configuration Field # >
```

Press 1 to enter a Database USER ID, and so on. Sample entries are illustrated in the following table:

Database USER ID	netranger@DBserver
Database PASSWORD	thePassword
Notify Person	john.doe@xyzcorp.com

After entering the local *sapd* tokens, press Enter to return to the Main Menu. From this menu, finalize the configuration by selecting the following:

```
10 - Generate Temporary Configuration Files
13 - Commit Temporary Configuration Files
```

At this point, please confirm the following information:

That this directory/file...	Contains the following...
/usr/nr/etc/daemons	nr.loggerd and nr.sapd
/usr/nr/etc/destination or /usr/nr/etc/smid.conf	nr.loggerd
/usr/nr/etc/loggerd.conf	defaults only
/usr/nr/etc/sapd.conf	defaults only

The nrConfig portion of the SAP installation for the NSX is now complete. The next step in the process involves using a command-line editor (like vi) to configure the default settings in both /usr/nr/etc/loggerd.conf and /usr/nr/etc/sapd.conf.

Configuring loggerd.conf involves setting the tokens that control serialization of log files by size (**NumberOfSwitchBytes**) and time (**NumberOfSwitchMinutes**), and setting **MinContextLevel**, a token that *loggerd* uses to filter context data out of event records.

After configuration of loggerd.conf, it is a good idea to generate some network activity that *sensord* will send as events. This will confirm that *loggerd* is properly configured to write Event and IP logs.

A key component of *sapd* is the management of the file system. sapd.conf contains the following default file management processes.

Process	sapd.conf File Management Triggers
Event Archiving	`FM_Action Archive_New /usr/nr/bin/sap/dump.sh $DirPath NEWLOG log.*`
IPLOG Archiving	`FM_Action Archive_Iplog /usr/nr/bin/sap/ipdump.sh $DirPath IPLOG`
Error Traps	`FM_Action Trap_Var /usr/nr/bin/sap/dump.sh $DirPath NRVAR errors.* messages.*`
Purging Dump	`FM_DirUsed DUMP_Purge 90 /usr/nr/var/dump Dump_Purge`
System Notifications	`FM_Action Ntfy_Urgent /usr/nr/bin/sap/notify.sh $DirPath $DirUsed $Notify1 URGENT`
	`FM_Action Ntfy_Purge /usr/nr/bin/sap/notify.sh $DirPath $DirUsed $Notify1 Purging`
	`FM_Action Ntfy_Warn /usr/nr/bin/sap/notify.sh $DirPath $DirUsed $Notify1 WARNING`

You may want to fine-tune the threshold parameters to better accommodate your disk space and system requirements.

Installing and Configuring SAP on a Director

Installation and configuration of SAP on a Director follows the same process as installation and configuration of SAP on an NSX sensor. Ensure that you ENABLE the Director instead of the NSX on nrConfig's Feature Selection Menu, so that it looks like the following:

```
FEATURE SELECTION MENU
Choose what features you want ENABLED on this host.
(Choosing an 'ENABLED' feature will disable it.)
        1 - NSX
ENABLED 2 - Director
ENABLED 3 - Logging
ENABLED 4 - Database Reporting
ENABLED 5 - File Management
        6 - Event Paging
ENABLED 7 - Postoffice Routing
        8 - Configuration Control
    Enter - CONTINUE
Feature # >
```

Follow the rest of the steps outlined in *Installing and Configuring SAP on an NSX* until you have finished configuration of loggerd.conf and sapd.conf.

When configuration of these files is complete, you are ready to set up the Director's DBMS functionality. You have the following data management options:

- Oracle DBMS
- non-Oracle DBMS

Oracle DBMS Setup

Oracle is the default SAP database. Setting up the Oracle database consists of the following steps:

1. **Database server setup.**
2. **Database schema setup.**
3. **Database load setup.**
4. **Database reports setup and customization.**

Database Server Setup

Please refer to Appendix D: DBMS REQUIREMENTS AND SETUP for more information on setting up the database server.

Database Schema Setup

Use the utility nrdb_master_create in /usr/nr/bin/sap/sql/skel/ to initialize your database schemas. You will be prompted for the following information:

- database user
- password
- location

Enter your database user name and password. For location, you have a choice of either local or remote. If you have a local DBMS, enter nothing at the `Remote DB Service` prompt. If you have a remote DBMS, enter the service name preceeded by an @ character.

Please refer back to the *Notification Interval, Conditions, and Destination* section on page 15 for examples of values to enter.

Database Load Setup

Use the SAP defaults unless you want to exclude elements. To do this, edit the load_run.sh file in /usr/nr/bin/sap and comment out those elements you wish to **include** in the reports. In the following example, the SAP_EXCLUDE_ALARM element has been commented out, and will therefore be included in generated reports. The other elements will be excluded from the reports.

```
#export SAP_EXCLUDE_ALARM=
export SAP_EXCLUDE_TCPCONN=
export SAP_EXCLUDE_CONTEXT=
export SAP_EXCLUDE_ALARM_1=
```

When this portion of the Oracle DBMS setup is complete, it is a good idea to verify that *sapd* will autoload staging event logs. You can do this by using instrumentation (described in *How to Use Instrumentation* in the *SAP Tutorial*) to observe the autoloading after log data has been loaded by *sapd*, and when you run queries.

Database Reports Setup and Customization

The queries shipped with the SAP generate basic reports, which are accessed via three different .sql entry points:

- event.sql
- space.sql
- time.sql

In turn, each of these queries gives access to a number of different subordinate queries, such as event1.sql, event2.sql, and so on. The query you run depends on the answers you give to the initial (event/space/time) query. Basically, the greater the number of the subordinate query, the more detail is returned.

You can customize these queries to better meet your reporting needs by editing `event.sql`, `event1.sql`, and `event2.sql` in /usr/nr/bin/sap/sql. Below is the interactive component of an `event` query. User input is bolded.

```
SQL> @event
EVENT DIMENSION QUERIES
=======================================================
1    alarm level summary
2    alarm signature summary
3    alarm signature with string data summary
These queries summarize NSX alarms with a primary focus
on the EVENT dimension.
You can filter the data returned to you with the following
criterion:
            ORGANIZATION_NAME
            MIN_EVENT_DATE
Please enter your desired value at the following prompts:
select query (1,2,3)>1
organization name (%)>%
minimum event date (MM/DD)>5/1
```

The results of the above query follow:

```
event1: alarm level summary
organization_name = %
minimum_event_date = 1997/5/1
Org Name         Level      From  To    Count      Recent
---------------  ---------  ----  ---   ---------  -----------
Auto Mart        3          OUT   IN    33         05:01 15:08
Data Pros        5          IN    OUT   3          05:02 08:45
Net King         5          IN    IN    15         05:02 11:21
Net King         3          OUT   OUT   4          05:01 19:15
Net King         3          IN    IN    2          05:01 15:03
Net King         3          IN    OUT   79         05:02 09:28
WheelGroup       3          IN    IN    231        05:02 09:37
WheelGroup       3          IN    OUT   176        05:02 09:37
WheelGroup       2          OUT   IN    9110       05:02 12:15
WheelGroup       2          IN    IN    18         05:01 14:06
```

In the example, the alarm **level** activity from four **organizations** is profiled. **From** refers to the source of the alarm direction (in relation to the trusted internal network) and **To** indicates the destination of the alarm direction. The **Count** is the number of alarms that match the specific criteria for each line in the generated report. **Recent** gives a timestamp of the last alarm activity for that criteria.

To change the scheduling of reports, customize the SAP triggers. These triggers can be found in /usr/nr/etc/sapd.conf.

SAP reports are written to /usr/nr/var/db. Each file contains a header with selection criteria and datestamp. You can customize the reporting process in several ways:

- The files DAY, WEEK, and MONTH in /usr/nr/bin/sap/sql are templates for report batches. They can be altered with various criteria to generate different sets of reports.
- New templates can be added to supplement the daily, weekly, and monthly templates. This will require a simple trigger modification or addition in sapd.conf.
- The actual SQL DML files (event1.sql, event2.sql, etc.) in /usr/nr/bin/sap/sql can be changed to alter the selection, sorting, summarization, and display of data.

Non-Oracle DBMS Setup

If you are setting up a generic SQL-based RDBMS, follow these steps:

1. **Database server setup.** Please refer to Appendix D: DBMS REQUIREMENTS AND SETUP for more information on setting up the database server.
2. **Database schema setup.** Follow the same procedure outlined in the *Oracle Database Schema Setup*, except the files for setup reside in /usr/nr/bin/sap/sql/skel.
3. **Database load setup.** Setting up the database loader involves choosing a loader (*sapx*, Oracle's own loader, or another load utility), and customizing the triggers in /usr/nr/etc/sapd.conf that schedule loading.
4. **Database reports setup and customization.** Follow the same procedure as with *Oracle Database Reports Setup and Customization.*
5. **Scheduler setup.** Ensure that the DBLoad and BatchReports tokens are properly set (in /usr/nr/etc/sapd.conf) to meet your needs.

If you are setting up a Remedy ARS trouble ticketing system, follow these steps:

1. **Copy the** `load_run.sh.remedy` **file from the /usr/nr/bin/sap/skel directory into /usr/nr/bin/sap, and rename it** `load_run.sh` **from /usr/nr/bin/sap.**
2. **Ensure that the DBLoad token is properly set in /usr/nr/etc/sapd.conf.**
3. **For logon requirements, edit the following tokens in /usr/nr/etc/sapd.conf: DBUser2, DBPass2, DBAux1, DBAux2, and DBAux3.**
   - DBUser2—Enter the Remedy user name (must not contain spaces).
   - DBPass2—Enter the Remedy user password (cannot be blank).
   - DBAux1—Enter "C" for this value (refers to Remedy's LANG variable).
   - DBAux2—Enter the name of the server running Remedy.
   - DBAux3—Enter NR_Alarm_Schema.
4. **To create a schema, use the Remedy GUI to import the** nr_alarm_schema.def **file from the /usr/nr/bin/sap/skel directory.**

SAP Tutorial

The goal of this section is to introduce you to **instrumentation**, **triggers**, and **polling intervals**. The section after this, *SAP Reference*, will build on the knowledge gained in this tutorial.

How to Use Instrumentation

The first thing you'll want to learn is how to use the SAP instrumentation to perform system monitoring of the following:

- Management
- File System
- Actions

You can use instrumentation to view the following: a summary of the FileMgmt token settings, a summary of the /nr/var directories, or the current configuration of actions.

Viewing FileMgmt Instrumentation

There are two ways to view FileMgmt status: Brief mode and Full mode. Brief mode provides a summarized view of the FileMgmt information, and includes overall status and uptime, a summation of errors, and a trigger history. Full mode is a superset of Brief mode, and also includes Scheduler status, normal summation, and trigger configuration.

To set for Brief mode, type the following command:

```
nrset 10007 . . 1 DisplayMode Brief
```

For Full mode, simply replace `Brief` with `Full` in the above command line.

To see what Brief mode instrumentation looks like, set your mode to Brief, and type in

```
nrget 10007 . . 1 FileMgmt
```

Something like the following information is returned:

```
Overall Status = ERROR Archive1 ;
SAPD up: 1 minute 2 seconds
current time: 05/12 11:10
-------------------------------------------------------------------
ERR total:1  last: 05/12 11:10  Archive1
-------------------------------------------------------------------
Trigger History
Status   OK    Error  Trigger Name   Action        Last Run
-------- ----- -----  -------------- ------------  -----------
Normal   1     0      OracleLoad      DBLoad_Proc   05/12 11:09
ERROR    0     1      Archive1        Archive_Proc  05/12 11:10
Normal   0     0      Batch_Reports   Batch_Proc
```

Following is an example of the information that would be returned if the mode were set to Full. Those sections that are in Full mode and not Brief mode have been bolded.

```
Overall Status = ERROR Archive1 ;
SAPD up: 4 minutes 0 second
current time: 05/12 11:13
PollingDelay = 1   DisplayMode = Full
Master Processing: IDLE
Batch Processing
(Batch_Reports) (pid 3246) (running 10 seconds)

ERR total:4  last: 05/12 11:13  Archive1

NORMAL  total:6  last: 05/12 11:09  OracleLoad

Trigger History
Status  OK    Error  Trigger Name    Action        Last Run
------- ----- -----  --------------- ------------  ------------
Normal  1     0      OracleLoad      DBLoad_Proc   05/12 11:09
ERROR   0     4      Archive1        Archive_Proc  05/12 11:13
Normal  0     0      Batch_Reports   Batch_Proc    05/12 11:11

Trigger Configuration
Trigger Name     Action        Condition
---------------- ------------  --------------------------------
OracleLoad       DBLoad_Proc   DirFiles = 1 in /usr/nr/var/new
Archive1         Archive_Proc  IdleTime(60) next: 05/12 12:13
Batch_Reports    Batch_Proc    CronTime     next: 05/14 21:00
```

Both Brief and Full modes contain an `Overall Status` line (the 1st line in the above example). This line indicates that *sapd* keeps track of its overall status; that is, if a launched action returns indicating an error condition, *sapd* knows that an error has occurred. In fact, the above example shows that an error has occurred:

```
Overall Status = ERROR Archive1 ;
```

If an error happens, *sapd's* status will remain in an ERROR state until one of the following occurs:

1. **The action is launched again and it returns a Normal (non-error) indicator.**
2. **The user executes the token UserStatus with the value "OK."**

For example, let's say that right before a scheduled staging of files to a database, the server goes down. If the server is down, then the staging does not occur at the scheduled time, and this would cause an Error to appear in the Overall Status line.

However, if the server is brought back on line before the next scheduled staging of data, and the staging occurs, then the Error would be replaced by a Recover in the Overall Status line, like so:

```
Overall Status = Recover Archive1 ;
```

Once this problem has been investigated and rectified, you can simply reset all of the status instances by typing the following command:

```
nrset 10007 . . 1 UserStatus OK
```

Once reset, the Overall Status line will look like the following:

```
Overall Status = Normal
```

Using VarSize

The VarSize token allows you to quickly assess the state of the SAP file system on an NSX or Director. Unlike FileMgmt, VarSize has only one mode of operation. To execute it, use the following command line:

```
nrget 10007 . . 1 VarSize
```

This will display a summary of all the /nr/var directories, which will look something like the following:

```
filepath          files   bytes    oldest         newest
/usr/nr/var       4       168782   05/07 16:10    05/12 11:09
/usr/nr/var/new   0       0
/usr/nr/var/old   53      354256   05/06 18:05    05/12 10:49
/usr/nr/var/iplog 0       0
/usr/nr/var/dump  0       0
```

Using FM_Action

The FM_Action instrumentation shows the current configuation of actions. To execute it, use the following command line:

```
nrget 10007 . . 1 FM_Action
```

This will display the following:

```
Action Configuration
Action          system command (args)
--------------  ------------------------------
DBLoad_Proc     DBLoad ($FileOldest )
Batch_Proc      /usr/nr/bin/sap/batch1 ($DirPath $Notify1 $DirUsed)
Archive_Proc    /usr/nr/bin/sap/test1 (ARG1 $DirPath )
```

Now that you know how to use SAP instrumentation, the next step in the tutorial is learning how to use triggers.

Triggers

sapd bases all of its actions on its trigger configuration. A trigger consists of a **condition** and an **action**. The condition defines **when** to launch, the action is **what** to launch. During the course of this discussion, you'll learn how to create and customize a trigger.

As stated earlier in the *File Management Thresholds* section, there are 5 different types of conditions used to initiate a launch; three are filesystem-based, and two are time-based, as illustrated in Table 5.2.

Table 5.2: Conditions

Basis	Name	Token	Meaning
Filesystem	DirFiles	FM_DirFiles	Number of files in location
Filesystem	DirSize	FM_DirSize	Bytes consumed in location
Filesystem	DirUsed	FM_DirUsed	Percentage of disk space used
Time	IdleTime	FM_IdleTime	Relative time (elapsed time) in minutes
Time	CronTime	FM_CronTime	Absolute time (depends on day & time) in minutes

The command-line syntax for setting a condition is as follows:

```
FM_<token>   LABEL   VALUE   LOCATION   ACTION
```

Where **LABEL** is a user-defined name; **VALUE** represents either number of files, number of bytes, percentage of disk space used, or time in minutes; **LOCATION** indicates a directory; and **ACTION** specifies the name of the action to be run.

For example, you could set the following condition:

```
FM_DirFiles  OraLoad  2  /usr/nr/var/new  OracleLoad
```

This means that when the number of files in the directory /usr/nr/var/new reaches two, then launch the action OracleLoad. As you can see, the VALUE placeholder (the "2" in our example) maps to the Meaning column in the above table.

After setting a condition, you'll need to set a corresponding action. The command line is very much like that for setting a condition, except you'll more than likely be using variables as your arguments. The syntax for setting an action is as follows:

```
FM_Action   ACTION  EXEC [ARGS]
```

As an example, type in a command that will set a corresponding action to the condition created above:

```
FM_DirFiles   OracleLoad   DBLoad $DirPath $FileOldest
```

The **ACTION** is a user-defined name, and should match the ACTION variable in the Condition for the trigger to work properly. The **EXEC** is the script that is executed—in the case of our example, it is DBLoad. The **ARGS** (or arguments) are literal parameters, unless you use the reserved macros listed in Table 5.3. In the case of our example, `$DirPath` will be replaced by the /usr/nr/var/new of the Condition, and `$FileOldest` will look for the filename of the oldest file in that directory.

You have just created your first trigger. Let's go over what the trigger does:

1. **The Condition stipulates that OracleLoad should run only if the /usr/nr/var/new directory contains two files.**

2. **The Action stipulates that OracleLoad will look in the /usr/nr/var/new directory for the oldest file, and perform a DBLoad.**

For more information on DBLoad and other *sapd* Actions, please refer to page 10.

Table 5.3: Macros and Expanded Values

Macro	Expanded Value
Filesystem Macros	
$DirFiles	count of all files in trLoc
$DirSize	size in bytes of all files in trLoc
$DirUsed	disk percentage used where trLoc resides
$DirPath	directory location
$FileOldest	filename of oldest file in trLoc
$FileNewest	filename of newest file in trLoc
Token Value Macros	
$Notify1	token value for NotifyPerson
$Notify2	token value for NotifyPerson2
$DBUser	token value for DBUser
$DBPass	token value for DBPass
Date/time Macros	
$TimeStamp	standard date/time stamp *(19970531.0137)*
$CurrTime	current time string *(23:37)*
$Today+1	date of tomorrow *(1997/05/31)*
$Today	date of today
$Today-1	date of yesterday
$Today-2	date of day-before-yesterday
$Today-3	date of day-before-the-day-before-yesterday
$Week	date of day a week ago
$Week-1	date of day 2 weeks ago
$Month	date of day starting this month
$Month-1	date of day starting the previous month

How to Configure a Trigger

Configuring a trigger to the needs of your organization is a three-step process:

1. **Edit the tokens in /usr/nr/etc/sapd.conf. They are easily recognized because they all begin with FM_ (e.g., FM_DirFiles, FM_DirSize).**

2. **After you've edited sapd.conf, either perform an nrstart (if you're not currently up) or an nrhup (if you're already up—this will activate the edited values in sapd.conf).**

3. **Finally, perform an nrget on both FileMgmt (in either Brief or Full mode) and FM_Action to view your settings and ensure that everything is correct.**

How to Change the PollingDelay

The speed at which *sapd* evaluates its triggers is controlled by the PollingDelay token. The valid values range from 1 to 9. The lower the value, the faster the polling. A value of 1 means *sapd* will evaluate the triggers every 10 seconds; a value of 9 means *sapd* will evaluate the triggers every 90 seconds.

If you want *sapd* to launch actions as quickly as possible, set the PollingDelay to 1. Use this if you have many small files in the /usr/nr/var/new area and you want them loaded quickly. If you want *sapd* to slow down launching of actions, set the PollingDelay to a larger value.

Use the following command-line syntax to change the PollingDelay on the fly:

```
nrset PollingDelay <1-9>
```

To set the PollingDelay permanently, edit sapd.conf, simply type the following:

```
PollingDelay <1-9>
```

SAP Reference

This section builds on the knowledge contained in the *SAP Tutorial* section. If you have not read the tutorial section yet, please do so now.

This section covers the following information:

- Thresholds and Triggers
- *sapr*
- Scripts and skel
- Operations

Thresholds and Triggers

This subsection discusses various ways to tune the thresholds in the /usr/nr/var file locations.

Remember that *loggerd* defines the PUSH threshold with **NumberOfSwitchBytes** and **NumberOfSwitchMinutes** tokens, and *sapd* defines various PULL thresholds with its filesystem and date/time condition tokens.

Do you have enough space?

The /usr/nr/var area must be big enough to include the data in the ~/new, /old, /dump, and /iplog directories, as well as a padding space of 100 megabytes. This padding is set as a default; the needs of your system may require more or less padding space to accomplish file management and push-pull processes.

The following formula is presented as a checksum to ensure that enough file space has been allocated:

```
nr/var size  > varTrap + (newTrap | newLoad || newArch)(_____) +
iplogArch + dumpOffline(_____) + padding(100MB)
```

Do you have enough room left over for archiving?

The **DirSize** token identifies how much of a disk partition can be dedicated to logging before an archiving action (such as compression or file deletion) can occur. It is a good idea to set the **DirUsed** token to a variable low enough to ensure that enough space is left to accomplish archiving. For example, setting the **DirUsed** token to 90 percent on a one gigabyte disk partition may leave enough room (100 MB), but if you are using a 500 megabyte disk partition, you may want to set the **DirUsed** token to 65 percent.

How do you want your data batched?

If you want fast pulling of files from a directory, set **DirFiles** to 1. To slow down pulling, set the **DirFiles** to a higher number, like 10. This will effectively change how many different files get batched together. Similar effects can be had with time thresholds. Setting **IdleTime** to a lower number will speed up polling, and setting it higher will slow polling down.

sapr

Reports

The SAP queries can be run either interactively or in batch mode. "Interactive mode" is useful for the most up-to-date information and for specific queries, but they take longer to run. "Batch mode" supports the "Ready Reports" but will only include information present in the database at the time of report generation. Please refer to the *Database Reports Setup and Customization* section for a sample interactive query and the results.

Customizing Queries

The SQL source code can be found in /usr/nr/bin/sap/sql/skel. This includes the queries that initialize the database schema for NetRanger tables as well as the queries used to analyze NetRanger data. Table 5.4 lists the included queries, which serve as starting points for query building.

Table 5.4: SQL Queries

Space Dimension Queries (space.sql)	
space1.SQL	source signature summary
space2.SQL	destination signature summary
space3.SQL	connection signature summary
space4.SQL	connection signature with string data summary
Time Dimension Queries (time.sql)	
time1.SQL	timeview signature summary
time2.SQL	timeview source signature summary
time3.SQL	timeview destination signature summary
time4.SQL	timeview connection signature with string data summary
Event Dimension Queries (event.sql)	
event1.SQL	alarm level summary
event2.SQL	alarm signature summary
event3.SQL	alarm signature with string data summary
System Queries (system.sql)	
system1.SQL	age and count of records in each table
system2.SQL	signature: name-number (interactive only)
system3.SQL	organization: name-number (interactive only)

SQL Database Information

Default Schemas

The native Oracle schema supported by *sapx* and *sapr* is illustrated in the following diagram. Each box on the left identifies how event data is grouped prior to being loaded into a database. Each box on the right represents the target database tables. The schemas for these tables are defined in Tables 5.5 through 5.9.

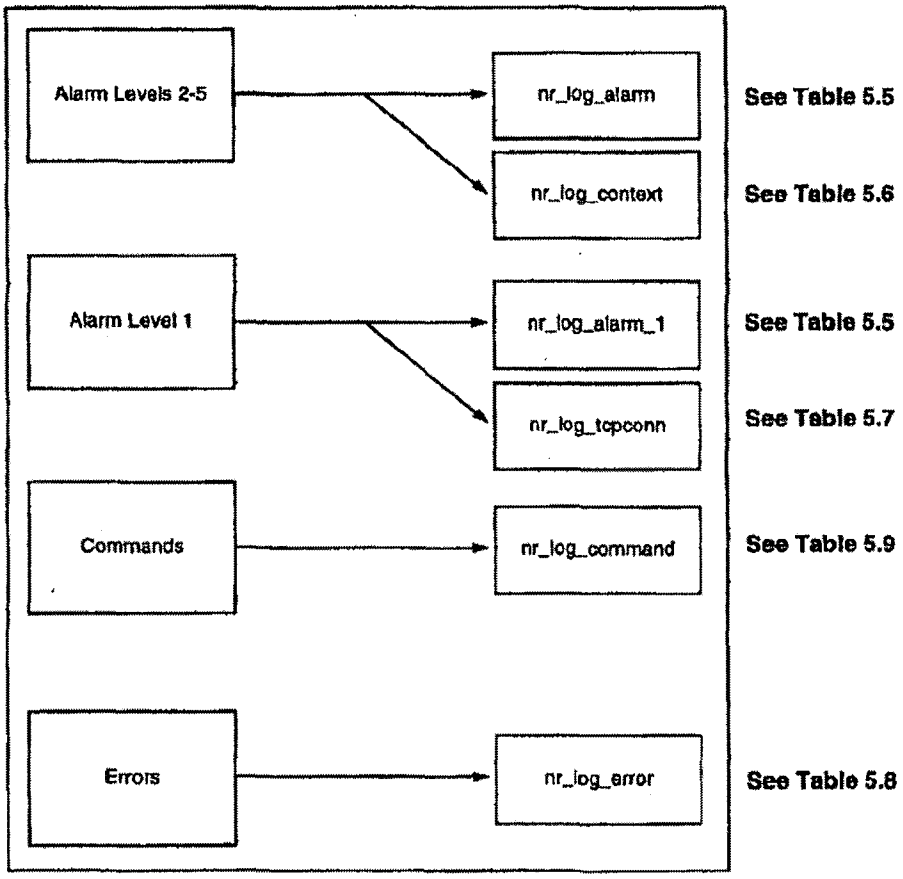


Table 5.5: nr_log_alarm and nr_log_alarm_1

Name	Type
EVENT_PROTOCOL	NUMBER(5)
RECORD_ID	NUMBER(10)
EVENT_DATE_GMT	DATE
EVENT_DATE_LOCAL	DATE
APP_ID	NUMBER(5)
HOST_ID	NUMBER(10)
ORG_ID	NUMBER(10)
FROM_STATE	CHAR(1)
TO_STATE	CHAR(1)
EVENT_LEVEL	NUMBER(5)
IP_SIGNATURE	NUMBER(10)
IP_SUB_SIGNATURE	NUMBER(10)
NETWORK_PROTOCOL	CHAR(3)
SRC_IP_ADDR	CHAR(32)
DST_IP_ADDR	CHAR(32)
SRC_PORT	NUMBER(5)
DST_PORT	NUMBER(5)
ROUTER_IP_ADDR	CHAR(32)
DATA_ALARM	CHAR(64)

Table 5.6: nr_log_context

Name	Type
RECORD_ID	NUMBER(10)
EVENT_DATE_GMT	DATE
HOST_ID	NUMBER(10)
ORG_ID	NUMBER(10)
DATA_MATCH	CHAR(64)
DATA_INCOM	VARCHAR2(768)
DATA_OUTGO	VARCHAR2(768)

Table 5.7: nr_log_tcpconn

Name	Type
RECORD_ID	NUMBER(10)
EVENT_DATE_GMT	DATE
EVENT_DATE_LOCAL	DATE
HOST_ID	NUMBER(10)
ORG_ID	NUMBER(10)
IP_SUB_SIGNATURE	NUMBER(10)
SRC_IP_ADDR	CHAR(32)
DST_IP_ADDR	CHAR(32)
SRC_PORT	NUMBER(5)
DST_PORT	NUMBER(5)
ROUTER_IP_ADDR	CHAR(32)

Table 5.8: nr_log_error

Name	Type
EVENT_PROTOCOL	NUMBER(5)
RECORD_ID	NUMBER(10)
EVENT_DATE_GMT	DATE
EVENT_DATE_LOCAL	DATE
APP_ID	NUMBER(5)
HOST_ID	NUMBER(10)
ORG_ID	NUMBER(10)
DATA_ERR	VARCHAR2(256)

Table 5.9: nr_log_command

Name	Type
EVENT_PROTOCOL	NUMBER(5)
RECORD_ID	NUMBER(10)
EVENT_DATE_GMT	DATE
EVENT_DATE_LOCAL	DATE
APP_ID	NUMBER(5)
HOST_ID	NUMBER(10)
ORG_ID	NUMBER(10)
SRC_APP_ID	NUMBER(5)
SRC_HOST_ID	NUMBER(10)
SRC_ORG_ID	NUMBER(10)
DATA_CMD	VARCHAR2(256)

Scripts and skel

The DBLOAD process relies on two types of scripts: control and conversion. Control scripts move data through the SAP system. Conversion scripts transform data into new output formats, such as cpio tape formats, printer hard copy, and relational database tables.

skel is a directory under /usr/nr that contains templates that specify alternate loading procedures for database types other than Oracle and Remedy ARS. These templates are effectively schema definitions for alternate bulk loaders.

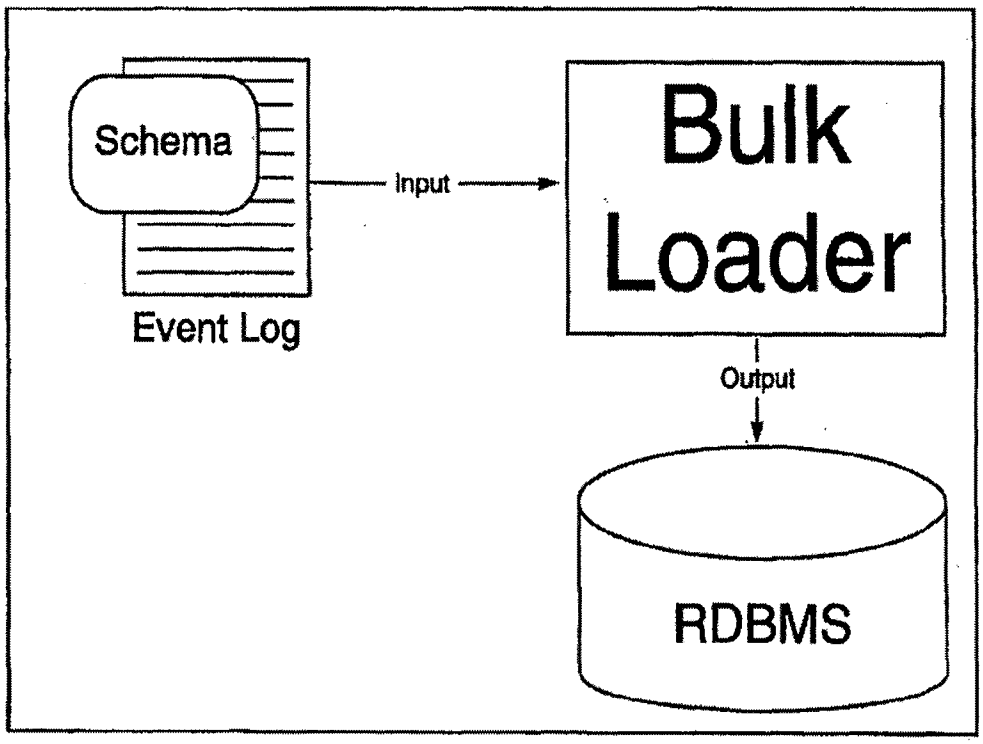


Figure 5.5: Bulk Loading Process

Alternate Loading Procedures

Defining what gets Loaded

sapx, by default, attempts to load error, command, and alarm record types. If you do not want one of these record types to be exported to your target database, modify the "for filetype in" command in the script /usr/nr/bin/sap/load_run.sh.

For example,

```
for filetype in 2, 3, 4
```

can be modified to receive only alarms by deleting the entry types for "2" (commands) and "3" (errors):

```
for filetype in 4
```

Changing the Target Database from Oracle to Remedy ARS

Replace the control script in /usr/nr/bin/sap/load_run.sh with the control script in /usr/nr/bin/sap/skel/load_run.sh.remedy.

Alternate Database Schemas and Destinations

Use the event log templates in /usr/nr/bin/sap/sql/skel as a guide for changing the schema of the database load process, or as a guide for exporting event log data to databases other than Oracle and Remedy ARS. Refer to the following for further information:

- Use the create_log*.SQL files as a template for the schema declaration.
- Use the oraldr_log*.ctl files as a template for SQL*Loader field mappings.
- Use the load_run.sh.oraldr as an example load_run.sh control script.

One of the requirements for the DBLOAD scripts is that they must return an exit status. Otherwise, *sapd* will not be able to determine when an error has occurred or perform an UNDO. Use the following settings for returning an exit status:

- If the script completed without error, return 0
- If an error was encountered during processing, return 1 or greater

NSOC Operations

Table 5.10 suggests scripts to help perform various NSOC-related tasks.

Table 5.10: Suggested scripts

Script	Task	Script Contents
Check Status		
sapdStatus	Display brief status	`nrset 10007 . . . DisplayMode Brief` `nrget 10007 . . . FileMgmt`

sapdStatusFull	Display full status	`nrset 10007 . . . DisplayMode Full`
		`nrget 10007 . . . FileMgmt`
		`nrget 10007 . . . VarSize`

sapdConfig	Show all configurations	`nrget 10007 . . 1 FileMgmt`
		`nrget 10007 . . 1 FM_Action`
		`nrget 10007 . . 1 DBConfig`
		`nrget 10007 . . 1 NotifyPerson`
		`nrget 10007 . . 1 NotifyPerson2`
		`nrget 10007 . . 1 NotifyInterval`
		`nrget 10005 . . 1 NumberOfSwitchBytes`
		`nrget 10005 . . 1 NumberOfSwitchMinutes`
		`nrget 10005 . . 1 MinContextLevel`

Make Adjustments		
sapdFast	Accelerate Polling	`nrset 10007 . . . PollingDelay 1`
sapdSlow	Decelerate polling	`nrset 10007 . . . PollingDelay 9`
sapdOK	Reset token status	`nrset 10007 . . . UserStatus OK`

sapdConfNew	Replace/ update configuration	`cp /usr/nr/etc/sapd.conf /usr/nr/etc/sapd.conf.prev`
		`cp /usr/nr/etc/sapdxxx.conf /usr/nr/etc/sapd.conf`
		`nrexec 10007 . . . ReadFileConfig`

sapdConfSave	Write active token files to sapd.conf	`nrexec 10007 . . . WriteFileConfig` `cat /usr/nr/etc/sapd.conf`
Serializing Log File		

loggerdSwitch	Switch, serialize, or execute DBLoad	`nrget 10007 . . . FileMgmt`
		`nrget 10007 . . . VarSize`
		`nrexec 10005 . . . SwitchFileLog`
		`nrget 10007 . . . VarSize`
		`sleep 60`
		`nrget 10007 . . . FileMgmt`
		`nrget 10007 . . . VarSize`

DBA Operations

Oracle database administrators are responsible for maintaining the SAP-default database. Some of the techniques described below will apply to other DBMS facilities. Once the database facilities are in place, the DBA typically will monitor the data areas, keeping operations smooth.

The DBA has two primary functions:

- Fix storage problems in the Oracle data files.
- Purge old data from database.

Fix Storage Problems

Table 5.11 illustrates recovery procedures for problems in the different Oracle data files.

Table 5.11: Recovery Procedures for Oracle Data File Problems

Area	Problem	Fix
DATA	Table storage overflow	• Extend DATA storage • Purge data
TEMP	Detailed query caused an overflow	• Extend TEMP segment • Tune TEMP segment • Filter more data from the selected DML
RBS	Purge overflows RBS	• Expand the size of your RBS • Refer to the PURGE section for alternate methods

Purging Data

Because of extended database operations, collected data will eventually outgrow the size of your Oracle disk partition. For this reason, you will need to occasionally purge the data from the nr_log_* tables.

The method you use to purge the data depends on your schema and the extent of your Oracle facilities (i.e., a large RBS will allow mass deletion, whereas a smaller RBS would require incremental and localized deletions).

A purge should not be wholesale—instead, selectively purge data from your tables. Save important alarm information and dispose only of the old or irrevelant data. Use the following guidelines as a basis for your purge strategy:

1. **First, generate some reports from the data you are ready to delete.**
2. **If you are logging Level 1 or Context data, purge these first.**
3. **Use the system1.SQL and event2.SQL queries to view the distribution of the data in your database. These queries will show the count of records in each table as well as signature-based distribution of records.**
4. **Selectively purge records from the nr_log_alarm tables.**

To remove **all** the records in a table:

1. **Drop the nr_log_alarm_1 table.**
2. **Use create_nr_log_alarm_1.sql to recreate the table.**

To remove **some** records based on time, use the following command line:

```
delete from nr_log_alarm where event_date_gmt < to_date('01/01','MM/DD')
```

To remove **some** records based on signature, use the following command line:

```
delete from nr_log_alarm where event_level < 4 and ip_signature = 8000
```

The above command lines can be combined to delete some time- and signature-based records. You can also use other fields, such as src_ip_addr (Source IP Address) to delete records from an individual machine. In other words, there is no need to use just time- and signature-based constraints in a purge, although they are the most commonly used.

NOTE

Remember to commit after the purge commands complete.

Do not use drop_nr_tables.sql for the purge because it drops **all** tables, essentially wiping your database clean.

An automated approach to data purge would be to schedule a Daily CronTime Batch trigger in *sapd* that enters sqlplus and executes report and purge DML actions.

File Management Tokens

Table 5.12: FileMgmt Tokens

Tokens	Default
Tokens for Directory Paths[a]	
LogFilePathNew	/usr/nr/var/new
LogFilePathTmp	/usr/nr/var/tmp
LogFilePathOld	/usr/nr/var/old
LogFileMask	log.
LogFilePathDump	/usr/nr/var/dump
LogFilePathIPLog	/usr/nr/var/iplog
LogFilePathVar	/usr/nr/var
Tokens for DBLOAD Launch Scripts[b]	
ControlPre	/usr/nr/bin/sap/load_pre.sh
ControlRun	/usr/nr/bin/sap/load_run.sh
ControlPost	/usr/nr/bin/sap/load_post.sh
ControlUndo	/usr/nr/bin/sap/load_undo.sh
Oracle Tokens[c]	
DBUser	username for Oracle access
DBPass	password for DBUser
Tokens for Remedy or other DBMS Default[d]	
DBUser2	username
DBPass2	password for DBUser2
DBAux1	auxiliary environment information
DBAux2	auxiliary environment information
DBAux3	auxiliary environment information

a Using these defaults is recommended, but they can be changed if necessary.

b These tokens can be changed, either permanently or on the fly to facilitate custom operations. See documentation for *configd* and Java configuration for information about token operation.

c These must be set before you can connect to Oracle. They are set to match whatever user/password was specified in the DBMS setup (create user).

d These tokens are included to provide extra flexibility for non-Oracle databases.

TROUBLESHOOTING



Overview

Sometimes things go wrong. Components often display confusing and unexpected error messages, or behave outside their normal parameters. Identification of these symptoms, their possible causes, and their possible solutions could potentially be a time-consuming and frustrating experience.

Ideally, the material in this appendix will provide a framework to help you identify and resolve routine problems. When problems are not so routine, this appendix will also help you by ruling out possibilities before you call WGC's Technical Support. This will ensure a faster and more beneficial resolution to your problem.

This Troubleshooting appendix consists of the following sections:

- Director Not Running
- Director Running
- Connectivity
- NSX
- Oracle
- Security Analysis Package

Director Not Running

Symptom	Possible Cause(s)	Possible Solution(s)
The following error message is displayed when trying to start the Director system from an HP Terminal session: `Cannot write message to Director, errno =2`	This error occurs when *smid* tries to write to a socket that does not exist. This may occur because the Director's nrdirmap application has not created its communication socket in /usr/nr/tmp because the OpenView user interface (ovw) was not started.	First ensure that the underlying NetRanger services, such as *postoffice*d and *smid* are running. The easiest way to accomplish this is to execute the following from the console command line: `/usr/nr/bin/nrstatus` Run /usr/nr/bin/nrstart as user netrangr if these services need to be started. Then bring up the OpenView user interface (ovw) by executing `$OV_BIN/ovw &`. This will automatically start nrdirmap.
The following error message is displayed when trying to start the Director system from an HP Terminal session: `Cannot write message to Director, errno = 233`	This error message is generated when *smid* writes to a socket whose buffer is overflowing. This can occur when the Director's nrdirmap application is not running.	Ensure that the OpenView user interface (ovw) is running by executing `$OV_BIN/ovw &`. This will automatically start nrdirmap.
The following error message is displayed when trying to start the Director system from an HP Terminal session: `Cannot write message to Director, errno = 239`	This error message occurs when *smid* and nrdirmap do not have adequate permissions to talk to one another via sockets in /usr/nr/tmp.	Ensure that the *smid* process is owned by user netrangr, and that nrdirmap runs as suid netrangr.

One of the following error messages is generated by HP OpenView upon startup:
`848967818198: WgcSema can't access`
`'/usr/nr/etc/nrdirmap.semaphore'`
`Error: nrdirmap:main, semaphore initialization failed.`
	The nrdirmap.semaphore file in /usr/nr/etc has been deleted or has improper permissions.	Ensure that /usr/nr/etc/nrdirmap.semaphore exists and that the netrangr *user* and group accounts have read access to the file. If the file does not exist, you can create the file with an ASCII editor, such as *vi*. Add a single character to the file and save it in /usr/nr/etc with the name nrdirmap.semaphore.
		Also ensure that /usr/nr/bin/nrdirmap runs as suid netrangr.

Symptom	Possible Cause(s)	Possible Solution(s)

nrdirmap: fatal: libovw.so.1: can't open file: errno=2	The LD_LIBRARY_PATH environment variable is not set properly in your user environment. Note that this may indicate that you are logged onto the Director platform as the wrong user.	Follow the instructions in Chapter 3 for setting up an HP OpenView environment for user accounts other than netrangr.
		
		If the user account is based upon either the bourne or korn shell, the following lines should exist in the user's $HOME/.profile:
		
		`if [ -d /opt/OV ] ; then`
		`. /opt/OV/bin/ov.envvars.sh`
		
		`PATH=$OV_BIN:$PATH`
		`export PATH`
		
		`LD_LIBRARY_PATH=$OV_LIB:$LD_LIBRARY_PATH`
		`export LD_LIBRARY_PATH`
		`fi`
		
		If the user must use a shell other than ksh, then the lines above must be translated into the appropriate scripting language and placed in the appropriate startup file.

Director Running

Symptom	Possible Cause(s)	Possible Solution(s)
The Director's security map contains an NSX icon, but fails to show any events for that NSX.	The Director's **Severity Status** attributes are set higher than the level of alarms being generated by the NSX.	From the Director interface, highlight the icon for the NSX system and then either press **Ctrl+O** or select **Edit→Describe→Modify** from the menu bar. Then select **NetRanger/Director** and press **View/Modify**. Ensure that the *Minimum Marginal* and *Minimum Critical* status thresholds are low enough to register events from the NSX in question.

	The level of alarms generated by the NSX system fall below the **routing threshold** set in the /usr/nr/etc/destinations file.	If the Director Severity Status thresholds are set properly, ensure that the routing threshold in the NSX's /usr/nr/etc/ destinations file is set low enough to route information to the Director. In the following example, the destinations file has the Director's *smid* event level set to "4", which means that only alarm events of level 4 and 5 will be forwarded onto the Director platform running on director1.wheelgroup:
		`1 datastore.wheelgroup loggerd 1 EVENTS,ERRORS,COMMANDS`
		`2 director1.wheelgroup smid 4 EVENTS`
		The Director should begin displaying alarms properly if you reset datastore's alarm threshold from "4" to "2".

Symptom	Possible Cause(s)	Possible Solution(s)
An NSX's alarms are properly displayed on the Director security map, but information on those alarms does not appear in the **Show Current Events** window **and** the event log file in the Director's /usr/nr/var directory does not contain any records from that NSX.	The Director *loggerd* daemon is not listed as a destination in either the NSX's or the Director's configuration files.	You can either create an entry in the NSX's /usr/nr/etc/destinations file for the Director's *loggerd* service, or you can create a **DupDestination** entry in the Director's /usr/nr/etc/smid.conf file to redirect event data to *loggerd* from *smid*. The latter would look like this: `DupDestination <HostId>.<OrgId> loggerd 1 EVENTS,ERRORS,COMMANDS` Where `<HostId>.<OrgId>` are the Director's values (e.g., director1.wheelgroup). **Note:** The first option requires that the NSX's destination file contain two entries for the Director system: one for *smid* and another for *loggerd*. This doubles the amount of network traffic that must be sent to the Director. Although the second option will create some overhead for *smid*, it is the preferred configuration, because it only requires one copy of event data to be sent over the communication pathway.
Information is properly displayed in the Director's **Show Current Events** window, but the mouse pointer turns into an hourglass and never changes back.	The current events utility continues to pull information from the Director log files as long as the window is up.	Press the **Stop** button to terminate the filtering application. You can then use the scrollbars and menu options to look at the data. Press the **Close** button to exit this window.

Connectivity

Symptom	Possible Cause(s)	Possible Solution(s)
Unable to access an NSX sensor or any of the NetRanger services running on the system.	Can you ping the NSX Sensor? **Yes**—This means that the NSX Sensor is powered up but that something is wrong with the NSX services. **No**—The BorderGuard is shunning the Director. This can occur when: • an attack is run from the Director system (as part of a system test or demonstration) that causes the NSX to shun the Director system. • the NSX does NOT contain a **`NeverShunIPAddress`** entry for the Director system.	Telnet onto the box as user netrangr and run **`/usr/nr/bin/nrstop`**. Try to deduce why services were disrupted by examining whatever error files exist in /usr/nr/var. If errors are found, refer to the appropriate Troubleshooting section, or contact WheelGroup Technical Support. Once the cause has been identified, restart the NSX services by executing **`/usr/nr/bin/nrstart`**. Gain access to the NSX system to either issue an unshun command via local execution of nrset, or restart *managed*, which will clear all current shuns on the BorderGuard. In either case you need to gain access to the NSX sensor via one of the following means: • connect to the NSX sensor via its modem • connect to the NSX via its COM1 serial port • connect to the NSX system's BorderGuard via the BorderGuard's console Although the first option provides remote access to the NSX system, the second and third options require physical access to the NSX. Option b also requires the use of a notebook computer or a terminal device.

NSX

Symptom	Possible Cause(s)	Possible Solution(s)
Unable to start or stop the NetRanger daemon processes when running `nrstart` or `nrstop`.	You are trying to run these utilities from an account that does not have access rights to the NSX daemons.	Ensure you are logged onto the NSX or Director systems under the same user account that was used to start its daemon services. (The default is user netrangr.)
/usr/nr/var/errors.smid contains multiple instances of the following error message: `socket received data from unknown host.`	The NSX system's *sensord* is receiving `copy_to` messages from a BorderGuard or Passport security device that has not been properly defined in /usr/nr/etc/sensord.conf.	Properly define a **RecordOfDataSource** for the BorderGuard or Passport device that is trying to send `copy_to` messages to the NSX system. Refer to Chapter 3 for information on properly defining the `<IP address>` and `<IP mask>` values for this sensord.conf token.

Oracle

Symptom	Possible Cause(s)	Possible Solution(s)
Cannot determine if Oracle is installed.		Check local and mounted file systems using commands `df`, `mount`, and `find`. Look for `oracle` and `product/7*`.

Cannot determine if Oracle is running.		Use the following command:
		`ps -ef | grep ora`
		After entering the command, the following should appear:
		`oracle 360 1 0 Jun 14 ? 0:01 ora_pmon_NRDB`
		`oracle 362 1 0 Jun 14 ? 4:36 ora_dbwr_NRDB`
		`oracle 364 1 0 Jun 14 ? 3:13 ora_lgwr_NRDB`
		`oracle 366 1 0 Jun 14 ? 0:40 ora_smon_NRDB`
		This means that there is an Oracle instance of SID=NRDB running since June 14. The user account oracle is the owner of these processes.

Oracle Installer (orainst) was unable to find any products to install.	**`start.sh`** was not run prior to starting orainst.	Run start.sh to prepare your environment for the orainst program. Look for the following file: `/cdrom/cdrom0/orainst/start.sh`

`sqlplus: not found`
or
`sqlldr: not found`
	The Oracle bin directory is not present or specified properly in your `$PATH`.	Set `$PATH` to include `$ORACLE_HOME/bin`.

The following error message is displayed when you try to run sqlplus: `~~~/oracle/product/7.3.2/bin/sqlplus: cannot open`	The shell finds sqlplus, but it cannot be executed. This can occur when your `$PATH` includes references to the wrong versions of the Oracle binaries. For example, you have mounted the wrong Oracle directories from a file server. Therefore, you are trying to execute HPUX binaries on a SPARC system.	Ensure that the `$ORACLE_HOME` directory contains the proper binaries for the platform you are running. Refer to the section entitled *Remote Setup: heterogeneous/homogeneous* in Appendix D.

Symptom	Possible Cause(s)	Possible Solution(s)

sqlplus, sqlldr, or *sapx* fail with the following SID error message:
`ERROR: ORA-01034: ORACLE not available`
`ORA-07200: slsid: oracle_sid not set.`
	The `ORACLE_SID` environment variable, which identifies which database **instance** to use, was not set properly prior to starting sqlplus.	Set `ORACLE_SID=NRBIG`.

sqlplus, sqlldr, or *sapx* fails with the following libc error message:
`libc.so.xxx: can't do something`
	$ORACLE_HOME/lib is not part of the `LD_LIBRARY_PATH` environment variable.	Add `ORACLE_HOME/lib` to the `LD_LIBRARY_PATH` environment variable. If you are running either the bourne or korn shell ensure that your $HOME/.profile contains the following entries:
		`LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$ORACLE_HOME/lib`
		`export LD_LIBRARAY_PATH`

sqlplus, sqlldr, or *sapx* fail with the following TNS error message: `ERROR: ORA-12154: TNS: could not resolve service name`	Oracle cannot understand the name specified in your connect string.	Ensure that the Oracle file tnsnames.ora resides in its proper location (usually $ORACLE_HOME/admin/network) and that it is properly formatted. Then use the tnsping utility to test sqlnet connectivity to your remote database.

sqlplus, sqlldr, or *sapx* return a TNS or USER/PASSWORD error message.	Improper connect string.	Correct the syntax on the connect string. Connecting from the shell prompt, do one of the following:
		If specifying the password on the command line, type:
		`sqlplus <user>/<password>@<host>`
		Otherwise, type:
		`sqlplus <user>@<host>`
		and sqlplus will prompt for a password.

Security Analysis Package

Symptom	Possible Cause(s)	Possible Solution(s)
Instrumentation shows proper configuration, but no actions are performed.	If there are no FM_Action items in sapd.conf, then nrConfig did not properly copy files into sapd.conf due to upgrade from 1.2.x to 1.3.x, or because nrConfig was not run at all.	Manually copy the sapd.conf.nsx or sapd.conf.director from /usr/nr/etc/wgc/templates into /usr/nr/etc/, replacing sapd.conf. **Note:** The file /usr/nr/etc/wgc/templates/sapd.conf contains descriptions of the tokens used by *sapd*. This file does not contain any "real" token values. It is intended as a reference for setting triggers in /usr/nr/etc/sapd.conf
There are extraneous files in the /usr/nr/var directory structure, and there exists a /usr/nr/var/old directory.	You have upgraded from SAP 1.2.x to 1.3.x, which does away with the /usr/nr/var/old directory. The upgrade has also left behind many files in /usr/nr/var.	Clean the extraneous files, delete the /usr/nr/var/old directory, and archive the /usr/nr/var/dump directory.
SQL queries do not display data.	You did not enter % as a wildcard.	Use % as a wildcard in your SQL queries.
From the SQLPLUS prompt, entering @event1, @space1, @time1, or @system1 does not return proper data.	You are using the wrong command line parameter.	SAP 1.3.x requires that you enter either @event, @space, @time, or @system. You will be queried for the desired drill-down level (e.g., 1, 2, 3).
Queries do not display new signatures.	You have upgraded from SAP 1.2.x to SAP 1.3.x without updating the nr_sigs and nr_orgs tables.	Update the nr_sigs table with /usr/nr/etc/signatures and the nr_orgs table with /usr/nr/etc/organizations. Use the code at the end of `nrdb_master_create` for this purpose.

FILTERS AND BRIDGE CONFIGURATION B

This section lists and explains the settings for all of the filter (fil), command (cmd), and data privacy facility (dpf) files required to deploy the BorderGuard or Passport as part of an NSX system. Please note that many of the settings described in this section have been automatically set for you by the nrconfig script shipped with NetRanger under the /usr/nr/bin directory. This information is presented here in the event you need to manually edit these files. Refer to the appropriate NSG BorderGuard Reference Manual for additional information about these files.

Standard Files

The standard files that must be configured and installed on a BorderGuard device are as follows:

- startup
- filters.cmd
- sleeves.cmd
- first.fil
- incom(x).fil
- last.fil
- shun.fil
- sleeves.fil
- params.dpf
- pubkeys.dpf
- sleeves.dpf

These files are discussed on the following pages.

startup—This is the boot configuration file that is read each time the BorderGuard powers up or resets. This file performs these tasks:

- Names the device.
- Compiles and applies filters.
- Configures interfaces.
- Sets up route tables.
- Executes the files required to establish encrypted sleeves.
- Establishes a nameserver.
- Starts daemons such as *telnetd*, *gated*, and *snmpd*.

filters.cmd—This file compiles all the filters that enforce your security policies. It is called in the startup file via the following command:

```
@filters.cmd
```

This file contains a single line for each filter used on the BorderGuard. Most *filters.cmd* files contain at least the following three entries:

```
ns> compile_filter first.fil
ns> compile_filter incom1.fil
ns> compile_filter shun.fil
```

Filter Templates

These files enforce security policies as well as establish filtering on the BorderGuard/ Passport. Refer to the *NetSentry User Guide* for information on how to create and apply filters. When you are editing these filters, you may find it helpful to refer back to *Chapter 3*, which you used when you defined your security policy.

first.fil—This file identifies which network traffic should be copied to the NSX sensor. First, change the IP address of the NSX that follows the `copy_to` commands if it has not already been changed. Then change any string-matching requirements defined in sensord.conf.

For string matching to be effective, all packets for a given service (e.g. Telnet at port 23) must be copied to port 35398. By default, FTP (21), Telnet (23), mail (25), DNS (53), RPC (111), and WWW (80) packets are all copied to the tcp_dp and tcp_sp command lines in *first.fil*. To set up string matching on additional services such as FTP data (20), add the ports to *first.fil*:

```
# Copy certain TCP traffic
ip_protocol in (6)
tcp_dp in (21,23,25,53,80,111,20)
copy_to IP_ADDRESS_NSX 35398 break;
```

NOTE

The port number following the NSX is 35398 instead of 35399 as it appears in the other filter files. **This is the UDP port for intrusion detection. Do NOT change it!**

incom(X).fil—This file is the core of the NetRanger security system and it establishes the security policy defined in Chapter 3. Apply this filter to each network interface to establish a security policy. The general naming convention is `incom1.fil` for interface 1, `incom2.fil` for interface 2, and so on.

NOTE

The default templates assign `incom1` to the interface connected to your untrusted networks. Notice that the number following the `copy_to` command in this file is 35399 instead of 35398, as indicated in the previous filter file. **This is the security policy monitoring port. Do NOT change it!**

If nrconfig has not been run, the first information that should be included is the IP address of the NSX after every `copy_to` command.

```
copy_to IP_ADDRESS_NSX 35399 break;
```

Next, make changes to the *Incom(X)_ip_spoof_fail* filter. **This filter prevents and alarms on any connections from outside your trusted network that have a source address from your internal network.** When applied to your trusted network interface, it assures that no outgoing traffic is spoofing an external address. Collect all of your internal class C and class B addresses and create an entry for each one. You should also create an entry for the individual external IP address. In the case of a site with two internal protected networks with class C addresses of 199.99.99.0 and 199.99.100.0 and a router with an external IP address of 199.99.101.1, the filter attached to the untrusted network would look like the following:

```
filter INCOM1_IP_SPOOF_FAIL
ip_sa mask 0xFFFFFF00 in (199.99.99.*)
copy_to IP_ADDRESS_NSX 35399
icmp_unreachable fail;
ip_sa mask 0xFFFFFF00 in (199.99.100.*)
copy_to IP_ADDRESS_NSX 35399
icmp_unreachable fail;
ip_sa mask 0xFFFFFFFF in (199.99.101.1)
copy_to IP_ADDRESS_NSX 35399
icmp_unreachable fail;
end
```

For the filter attached to the trusted network, make the following changes:

```
filter INCOM1_IP_SPOOF_FAIL
not ip_sa mask 0xFFFFFF00 in (199.99.99.*)
not ip_sa mask 0xFFFFFF00 in (199.99.100.*)
copy_to IP_ADDRESS_NSX 35399
icmp_unreachable fail;
end
```

The next change you need to make is based on your security policy. Every allowed service should have a corresponding filter (for example, you should have a filter similar to `incom2_smtp_fail` for e-mail). You need to edit each of these filters in accordance with your policy, which typically is to allow or block network access. For each allowed service in the security policy, edit the current filter with its name. There are two basic cases for allowable services:

- **The first and simplest case is when you want to allow all services of this type through.** This is most often the case for the interface attached to a trusted network, because users want unrestricted access to e-mail and WWW services. For example, to allow all outgoing mail traffic, simply comment out the body of the filter *incom(X)_smtp_fail* as follows:

```
filter INCOM2_SMTP_FAIL
#tcp_dp in (25)
#copy_to IP_ADDRESS_NSX 35399
#icmp_unreachable fail;
end
```

  This allows all outgoing connections to port 25.

- **The second case for allowing a given service involves restricting that service to a specific destination or source IP address.** This type of restriction is usually applied to the interface attached to an untrusted network. In this case, the filter should be edited as follows:

```
filter INCOM1_ICMP_FAIL
ip_protocol in (1)
#This is where you should enter the typefields of the ICMP
#services you want to block. In this example, we are blocking
#"echo requests" type field 8. To block "timestamp requests"
#you would add 13. tl_byte 0 in (8,13)
tl_byte 0 in (8)
not ip_sa in (198.98.98.*)
not ip_da in (IP_ADDRESS1, IP_ADDRESS2, IP_ADDRESS3, ETC.)
copy_to IP_ADDRESS_NSX 35399
icmp_unreachable fail;
end
```

The above example allows all ICMP echo requests that are from the network 198.98.98.0 or that are destined for IP_ADDRESS1, 2, or 3. All of the filters should comply with your security policy requirements. Please refer to the *NetSentry User's Guide* for more information on this topic.

last.fil—Although this filter is not required, it provides a convenient way to permanently block a particular site. This type of filter might look like this:

```
filter LAST_BLOCK_FAIL
ip_sa in (BLOCKED_IP_NETWORK)
copy_to IP_ADDRESS_NSX 35399
icmp_unreachable fail;
end
```

shun.fil—NSX system uses this file to **dynamically block** outgoing as well as incoming IP addresses. It is not applied to a BorderGuard/Passport until the NSX generates an alarm for an attack signature that is configured to shun.

NOTE

The name of this file should NOT be changed. The only thing that needs to be edited in this file is the IP address of the NSX that follows the `copy_to` command. If you have run *nrconfig*, this should already be changed.

NetRanger and Bridging

Below is a sample startup file for a BorderGuard 2000 routing on four interfaces. First, the name is set, then the *filters.cmd* script is run (this compiles the NetSentry filters). The first_filter is then applied on the IP first filter point. The four interfaces are then started, followed by starting the Telnet daemon on the BorderGuard. Each of the interfaces has certain filters applied when it is configured. These filters are on the incoming/outgoing filter points for each interface. Each of the Ethernet ports is configured at startup by the file.

```
system set name border
@filters.cmd
netsentry apply ip first_filter on first
ip start if en01 10.1.1.1 netmask 0xffffff00 incoming=incom1_filter \
outgoing=out1_filter
ip start if en02 10.1.2.1 netmask 0xffffff00 incoming=incom2_filter
ip start if en03 10.1.6.1 netmask 0xffffff00 outgoing=out3_filter
ip start if en04 10.1.4.1 netmask 0xffffff00 incoming=out4_filter
ip start telnetd
```

If the same BorderGuard 2000 were deployed in bridge mode, the important difference is that it has only 1 IP address on 1 IP network. Again, the name is set first, followed by the script to compile NetSentry filters. Here, the first_filter is applied on the MACADDR first filter point. That is the filter point for bridging. NetSentry filters are applied on each of the interface's corresponding incoming/outgoing filter points by the `set` commands as shown in the example below.

```
system set name border
@filters.cmd
netsentry apply macaddr FIRST_FILTER on first
ip start if br01 10.1.1.1 netmask 0xffffff00
bridge set port en01 incoming=INCOM1_FILTER outgoing=OUT1_FILTER
bridge set port en02 incoming=INCOM2_FILTER
bridge set port en03 outgoing=OUT3_FILTER
bridge set port en04 incoming=OUT4_FILTER
ip start telnetd
```

Unlike configuring interfaces for IP routing, the bridge configuration is automatically stored on the BorderGuard file system as configuration commands are issued. Two files are created with the names BRIDGE.DAT and BRIDGE.HDR. The command to start an interface to bridge is

```
bridge set port <interface> state on
```

The BorderGuard will immediately configure the interface by turning the port off, then on, and finally saving the configuration to disk. Here is an example:

```
BG2000 Bridge> set port en01 state on
bridge: syslog contains unread text
bridge: do 'bridge show syslog' to view
BG2000 Bridge>
%Ethernet-8-2505, 18-Feb-1997 22:10:38.966
Ethernet port en01 is OFF.

%Ethernet-8-2504, 18-Feb-1997 22:10:38.966
Ethernet port en01 is ON.

%Bridge-8-322, 18-Feb-1997 22:10:39.260
Save state complete

BG2000 Bridge> sys dir
File name            Size      Date       Time
.                   <dir>  07-Feb-1997  12:08
README              29300  15-Feb-1997  15:30
PROFILE                21  09-Feb-1997  12:51
STARTUP                62  09-Feb-1997  12:58
SHUN.FIL               19  10-Feb-1997  22:34
FIREWALL.DEF        19224  15-Feb-1997  15:30
FIREWALL.SET         7180  15-Feb-1997  15:30
BRIDGE.DAT            848  18-Feb-1997  22:10
BRIDGE.HDR             44  18-Feb-1997  22:10

9/256 files, 57100/86660 bytes used/free
42608 recoverable bytes
BG2000 Bridge>
```

Once this command is issued for an interface, it does not need to be issued again—hence the exclusion from the startup file. However, if the command is placed in the startup file, it does not cause any problems.

NOTE

Do not assign an IP address to an interface if you want it to bridge IP packets. Routing takes precedence over bridging. If a port, like en01, is configured to do bridging and it has an IP address assigned to it, the IP packets are routed and other protocols (IPX/SPX, AppleTalk, etc.) are bridged.

Filters

In bridging mode, the following should occur for NetRanger filter files:

1. **copy_to should be replaced by log_to.**
2. **UDP port 35398 should be replaced by 35400.**
3. **UDP port 35399 should be replaced by 35401.**

Here is an example of a part of the filter netranger:

```
filter NETRANGER

        ip_protocol in (1)
log_to 10.1.100.1 35400 break;

ip_protocol in (6)
tl_byte 13 mask 0x07 in (1,2,4)
log_to 10.1.100.1 35400 break;

tcp_dp in (21,23,25,53,80,111,512,513,514)
log_to 10.1.100.1 35400 break;

tcp_sp in (21,23,25,53,80,111,512,513,514)
log_to 10.1.100.1 35400 break;
...
...
...
```

Here is an example of a security policy filter:

```
filter INCOM1_IP_SOURCE_ROUTE_FAIL
ip_option_present 0x83
log_to 10.1.100.1 35401
icmp_unreachable fail;
ip_option_present 0x89
log_to 10.1.100.1 35401
icmp_unreachable fail;
end
```

DPF Files

The BorderGuard 1000 and 2000 systems are equipped with a powerful encryption capability that can establish a Virtual Private Network (VPN) between geographically remote sites over public networks.

At a minimum, WheelGroup recommends that you use encrypted sleeves to encrypt all traffic between remote networks. It is not necessary to use encrypted links between internal NetRanger components whose traffic is only crossing internal protected private networks.

A good method to determine the necessity for encrypted sleeves is to trace the path of alarms and traffic traveling between two NetRanger components and consider the security of the networks that are crossed. If you feel comfortable with those networks, it is probably not necessary to use encryption and incur the performance reduction on traffic traveling within the sleeve.

Please note that the Passport system currently does not provide any type of VPN Facilities.

Refer to the NSG manual, *Data Privacy Facility User's Guide*, before attempting to make changes to any of the files in this section.

pubkeys.dpf—This file contains the public keys for the BorderGuard. This file should have an entry for the local system and each remote BorderGuard for which an encrypted sleeve is maintained.

NOTE

sleeves.dpf requires long keys. Be sure that you generate long public keys for both ends of the encrypted sleeve.

sleeves.dpf—This file contains sleeve definitions for the BorderGuard's DPF. During initialization, it is loaded as a set of sleeve definition commands and placed into the DPF interface's working storage. The default sleeve settings rarely need editing. The only entries that should be edited are sleeve definitions. The format for these entries is

```
sleeves_<orgId1>_<orgId2>_<sleeve_num> group1 <IP> group2 <IP>
```

where `orgId` corresponds to the organization ID established in the /usr/nr/etc/organizations file discussed in *Appendix E*. `orgId1` is the organization at one end of the sleeve and `orgId2` is the organization at the other end of the sleeve; `orgId1` may equal `orgId2`. The `sleeve_num` corresponds to the sleeve number at a particular site. The IP addresses assigned for group1 and group2 correspond to the router/bridge interface designated at the BorderGuard at each end of the sleeve as the DPF interface.

NOTE

Sleeve definitions should be identical at both the local and remote site.

For example, consider site #1 with two BorderGuards, which have the DPF IP addresses 199.99.99.1 and 199.99.100.1, and site #2 with one BorderGuard with a router DPF IP address 198.98.98.1.

The following sleeve definitions might apply:

```
sleeves_100_101_1 group1 199.99.99.1 group2 198.98.98.1
sleeves_100_100_1 group1 199.99.99.1 group2 199.99.100.1
```

This would establish two sleeves. The first is between one of the BorderGuards at site #1 and the BorderGuard at site #2. The second is between the two BorderGuards at site #1.

NOTE

Be sure to associate the correct IP address with the correct group (`group1` or `group2`).

sleeves.fil—This file contains two filters for every DPF sleeve connection defined in *sleeves.dpf*. The first filter controls outbound traffic and the second filter checks inbound traffic.

NOTE

The **IP address of the NSX** (after the *copy_to* statements) and the **sleeve name** must match the name designated in *sleeves.dpf*. For example, if site #1 has a class C address 199.99.99.0 and site #2 has a class C address 198.98.98.0, and the sleeve name is s_100_101_1, the filter at site #1 would look something like the following:

```
filter DPF_100_101_1
ip_sa in (199.99.99.*)
set_sleeve s_100_101_1;
end;
filter DPF_FAIL_100_101_1
ip_da in (199.99.99.*)
not sleeve s_100_101_1
icmp_unreachable
copy_to IP_ADDRESS_NSX fail;
end;
```

A corresponding filter would be created for the remote site and would replace 199.99.99.* with 198.98.98.* but the sleeve names would remain the same.

sleeves.cmd—This file compiles and applies all the filters required to establish encrypted sleeves between a local BorderGuard and one or more remote sites. The filters are established in the *sleeves.fil* file and should be applied so that all traffic traveling between two sites is encrypted. For example, if site #1 had a class C address of 199.99.99.0 and site #2 had a class C address of 198.98.98.0, then the apply statements would be similar to the following:

```
ip apply DPF_100_101_1 on 199.99.99.* to 198.98.98.*
ip apply DPF_FAIL_100_101_1 on 199.99.99.* to 198.98.98.*
```

In this example, `DPF_100_101_1` and `DPF_FAIL_100_101_1` correspond to the filter names established in the file *sleeves.fil*.

Bridging and DPF

DPF sleeves can be used when the BorderGuard is running in bridge mode. DPF sleeves should be defined and established normally. However, packets should be inserted into the sleeve using an IP filter. This is required because the set_sleeve action is not supported in filters on bridging filter points. To allow IP packets to pass through an IP filter on a BorderGuard running in bridge mode, follow these steps:

1. **Establish a normal DPF sleeve, correctly configuring** pubkeys.dpf, params.dpf, **and** sleeves.dpf.

2. **Assign the IP address assigned to port br01 as the DPF address of the BorderGuard.**

3. **Define a filter to place packets into the DPF sleeve (an example is shown below).**

   All the packets coming from the 10.1.6.x network and going to the host 10.1.9.10 will be placed into the DPF sleeve qa_test.

```
filter DPF
        ip_sa mask 0xffffff00 in (10.1.6.0)
        ip_da in (10.1.9.10)
                set_sleeve qa_test;
end
```

4. **Apply the filter on the IP first filter point. (The filter point** local_in **would also work.) An example (which would be added to the startup file) follows:**

```
ns apply ip DPF on first
```

5. **Verify that none of the bridge security policy filters fail any packets destined for the sleeve.**

6. **Configure the NSX (or Director) to route packets to the other destination through the IP address assigned to port br01. An example follows:**

```
route add 10.1.9.10 <port br01 IP address> 1
```

This technique works because the NSX sends IP packets to the Director that are encapsulated in Ethernet packets with a destination Ethernet address corresponding to br01 (because it is the router). The IP packet will first go through all of the bridge filter points and then through the IP filter points because it is going to br01. The filter defined in Step 3 then takes the IP packets and puts them into the sleeve for transmission to the other side. If an IP packet that should NOT go into the sleeve is sent to br01, br01 routes that packet according to its own routing table.

When the packet comes out of the sleeve on the other end, the routing table is consulted and determines that the IP packet should be transmitted through the br01 interface, which in turn causes it to be bridged out into the network to its final destination.

NetRanger Support for Passport Bridging

NetRanger Version 1.3.1 supports the Passport in bridging mode. Two additional UDP ports have been added to nr.sensord to process the specialized log_to packets from the Passport. The table below lists the UDP ports for nr.sensord.

Socket Name	Default Port #	Supports
NCI1	35398	Intrusion Detection
NCS1	35399	Security Policy Monitoring
NLI1	35400	Intrusion Detection
NLS1	35401	Security Policy Monitoring
NLI2	35402	Intrusion Detection
NLS2	35403	Security Policy Monitoring

Socket Name Key

N—Network Systems

C—copy_to packets

L—log_to packets

I—Intrusion Detection

S—Security Policy Monitoring

#—Version Number

For example, `NLI2` indicates that support for Network Systems log_to packets for intrusion detection (version 2) is enabled.

When you enable intrusion detection support for a UDP port, packets sent to that UDP port will be used for intrusion detection and logging. A filter should be placed on the NetSentry first filter (for IP and Bridging) point to copy_to/log_to packets to the NSX for analysis.

When security policy monitoring support is enabled for a UDP port, packets sent to that UDP port will be analyzed to determine the name of the NetSentry filter that performed the copy_to/log_to. In the following example, the filter will fail certain packets and send them to the NSX:

```
filter INCOM1_FINGER_FAIL
   tcp_connect_request
   tcp_dp in (79)
       log_to 10.1.1.1 35401
       fail;
end
```

In this case, *sensord* will obtain the filter name `INCOM1_FINGER_FAIL` from the log_to packet, attempt to find a matching RecordOfFilterName entry to get the filter name SUB-SIGID, look up the corresponding alarm levels set by SigOfFilterName, and then transmit an alarm as needed. The IP packet is all but ignored except to obtain source/destination addresses/ports.

The entries in /usr/nr/etc/sensord.conf for the above security policy filter would look like the following:

```
RecordOfFilterName  1011   INCOM1_FINGER_FAIL
SigOfFilterName     1011     0   3   3   3   3
```

In prior releases of NetRanger, *sensord* only supported the first two sets of UDP sockets. The NLI2 and NLS2 sockets have been added to support the log_to packet format generated by the Passport. This format is different than the one supported by the NLI1 and NLS1 sockets for BorderGuard log_to packets.

NOTE

Extra caution must be used when creating filters to send log_to packets to the NLI2 and NLS2 sockets when in bridge mode.

Caution is necessary because the log_to header does NOT specify the type of MAC header. A Passport with Ethernet, Token Ring, and FDDI interfaces that sends log_to packets to the NSX will include differing MAC headers. The NSX cannot easily determine which MAC type is included in the log_to packet.

To deal with this problem, *sensord* looks at the NetSentry name in the log_to header to determine the type of MAC header.

The NLI2 socket REQUIRES the modification of the traditional NetRanger first filter. The filter is now divided into three sections, each of which is used to process different types of MAC packets. This ensures that the log_to header will contain one of the following three filter names for packets received at the NLI2 UDP socket:

NETRANGER_BR_ETHER	Ethernet support
NETRANGER_BR_TOKENRING	Token Ring support (not tested)
NETRANGER_BR_FDDI	FDDI support (not tested)

Following is a sample breakdown of the new filters to be used on the Passport in bridge mode:

```
filter FIRST_BR_FILTER        <-- Applied at the bridge first filter point
   filter NETRANGER_BR;
   filter SHUN;
end
filter NETRANGER_BR
   filter NETRANGER_BR_ETHER;
   filter NETRANGER_BR_TOKENRING;
   filter NETRANGER_BR_FDDI;
end
filter NETRANGER_BR_ETHER
  not mac_media_type in (1)     <-- Return from filter if not Ethernet
      break;
   ip_protocol in (1)             <-- Traditional filter
      log_to 10.1.100.1 35402 break;
   ...
   ...
   ...
end
```

The NLS2 socket requires that all security policy filters have one of the following strings prepended to the filter name to ensure that the type of packets sent to the NSX via log_to have matching MAC headers. If one of these strings listed in the table below is not prepended to the filter name, *sensord* will ignore the log_to packet.

ETHER	Ethernet support
TOKENRING	Token Ring support (not tested)
FDDI	FDDI support (not tested)

NOTE

Filter names should not exceed 31 characters in length. In some cases, prepending one of the MAC formats listed above to certain default NetRanger policy filters will result in a filter name that exceeds the 31-character limit. In those cases, truncation will be necessary.

In the following example, the security filter has been rewritten for the NLS2 socket. In this example, the filter is applied on the incoming filter point of an Ethernet interface. This ensures that all log_to packets from this filter include Ethernet MAC headers. If this filter were at another filter point (first, last, apply_table, etc.), then it would have to be broken up in triplicate like the first filter shown on the previous page.

```
filter ETHER_INCOM1_FINGER_FAIL
   tcp_connect_request
   tcp_dp in (79)
       log_to 10.1.1.1 35403
       fail;
end
```

DEFAULT NETRANGER ALARM SETTINGS C

This section lists the default NetRanger alarm settings. These settings provide for near-maximum intrusion detection but are not configured to provide automatic response. For each of the ID signatures, string matches, transport events, and policy violations, you can turn on auto-logging or auto-shunning by editing *sensord.conf* or changing the settings through the nrConfigure GUI. The default levels will give you some idea of the usual level of threat for each of these items.

You may want to emphasize certain events at certain sites by increasing the alarm level on the events or by turning on an automatic response. You can also lower the reporting level on certain events to de-emphasize them.

NOTE

The SYN flood detector is configured to work on services that are set up for string match examination. If this is not appropriate for your installation, it may be changed in the file *first.fil* on the BorderGuard. Refer to *Appendix B* for information about editing filter files. Keep in mind that misconfiguring these settings may greatly increase the load on the sensor and its portion of the network bandwidth without providing an increase in protection. Please consult WheelGroup Technical Support for advice on this adjustment.

Table C.1: Intrusion Detection Signatures, Sorted by Default Alarm Level

Alarm Level				
1	2	3	4	5
IP options - Bad Option List	IP options - Record packet route	sendmail reconnaissance	Fragmented ICMP traffic	IP options - Loose source route
IP options - Timestamp	ICMP Source Quench	Archaic sendmail attacks	Large ICMP traffic	IP options - Strict source route
IP options - Provide s, c, h, tcc	ICMP Time Exceeded for a Datagram	DNS HINFO Request	smail attack	ICMP network sweep w/Echo
IP options - SATNET id	ICMP Timestamp Request	DNS request for all records	sendmail invalid recipient	ICMP network sweep w/ Timestamp
ICMP Echo Reply	ICMP Information Request (obsolete)	ypserv attempt	sendmail invalid sender	ICMP network sweep w/Address Mask
ICMP Unreachable	ICMP Address Mask Request	ypbind attempt	DNS Zone Transfer from other port	TCP port sweep
ICMP Redirect (change a route)	Unknown IP Protocol	yppasswdd attempt	RPC dump (rpcinfo -p)	Half-open SYN attack
ICMP Echo Request	HTML file has .url link	ypupdated attempt	rexd attempt	WWW phf attack
ICMP Parameter Problem on Datagram	HTML file has .lnk link	ypxfrd attempt	ICMP flood	WWW general cgi-bin attack
ICMP Timestamp Reply	HTML file has .bat link	Sendmail decode alias	WWW .url file requested	TCP Hijacking
ICMP Information Reply (obsolete)		UDP bomb	WWW .lnk file requested	UDP port scan
ICMP Address Mask Reply		mountd attempt	FTP CWD ~root	TFTP passwd file attempt
TCP connection records		FTP SYST command attempted	FTP Improper address specified	Normal SATAN probe

Table C.1: Intrusion Detection Signatures, Sorted by Default Alarm Level

Alarm Level				
1	2	3	4	5
UDP packet records		FTP Authorization Failure	FTP Improper port specified	Heavy SATAN probe
DNS Zone Transfer Request		Telnet Authorization Failure	WWW .bat file request	RPC port registration
		Rlogin Authorization Failure		RPC port unregistration
		POP3 Authorization Failure		Proxied RPC request
				IP options - IP Fragment Attack
				FTP SITE command attempted
				NETBIOS OOB data
				rlogin - froot
				Ident buffer overflow
				Ident newline
				Ident improper request

Table C.2: Intrusion Detection String Matches, Sessions and Alarm Levels

String Match	Session	Alarm Level
"+ +"	Telnet	2
"+ +"	rlogin	2
"RETR passwd"	FTP	4
"/etc/shadow"	Telnet	4
"IFS=/"	Telnet	5
"IFS=/"	rlogin	5
"/etc/shadow"	rlogin	4

Table C.3: TCP/UDP Events and Alarm Levels

Event	Alarm Level
TCP	
Connection request- tcpmux	1
Connection request- echo	1
Connection request- discard	1
Connection request- systat	1
Connection request- daytime	1
Connection request- netstat	1
Connection request- chargen	1
Connection request- ftp-data	1
Connection request- ftp	1
Connection request- Telnet	1
Connection request- smtp	1
Connection request- time	1
Connection request- whois	1
Connection request- dns	1
Connection request- gopher	1
Connection request- finger	1
Connection request- www	1
Connection request- link	1
Connection request- kerberos-v5	1
Connection request- supdup	1
Connection request- hostnames	1
Connection request- iso-tsap	1
Connection request- x400	1
Connection request- x400-snd	1
Connection request- csnet-ns	1
Connection request- pop-2	1
Connection request- pop3	1

Table C.3: TCP/UDP Events and Alarm Levels

Event	Alarm Level
Connection request- sunrpc	1
Connection request- uucppath	1
Connection request- nntp	1
Connection request- ntp	1
Connection request- netbios (port #s 137, 138, and 139)	1
Connection request- imap2	1
Connection request- NeWS	1
Connection request- xdmcp	1
Connection request- nextstep	1
Connection request- bgp	1
Connection request- irc	1
Connection request- imap3	1
Connection request- ulistserv	1
Connection request- exec	3
Connection request- login	3
Connection request- shell	3
Connection request- printer	1
Connection request- courier	1
Connection request- uucp	1
Connection request- pcserver	1
Connection request- kerberos-v4	1
UDP	
UDP traffic - echo	1
UDP traffic - discard	1
UDP traffic - daytime	1
UDP traffic - chargen	1
UDP traffic - time	1
UDP traffic - dns	1
UDP traffic - tftp	3

Table C.3: TCP/UDP Events and Alarm Levels

Event	Alarm Level
UDP traffic - gopher	1
UDP traffic - www	1
UDP traffic - kerberos-v5	1
UDP traffic - sunrpc	1
UDP traffic - ntp	1
UDP traffic - xdmcp	1
UDP traffic - bgp	1
UDP traffic - imap3	1
UDP traffic - ulistserv	1
UDP traffic - biff	1
UDP traffic - who	1
UDP traffic - syslog	1
UDP traffic - printer	1
UDP traffic - talk	1
UDP traffic - ntalk	1
UDP traffic - route	1
UDP traffic - nfs	1

Policy Violations

The sample policy filter *incom1.fil* is designed to identify incoming traffic and pass or fail it based on set conditions. For example, if you specified an FTP server and an SMTP server during the NetRanger configuration, the sample filter would be modified to allow these hosts as exceptions to the rules which fail incoming FTP and SMTP. This sample policy is excellent for use if the NetRanger is protecting the access point between a LAN and an untrusted WAN. If the NetRanger is providing separation between the Finance and Engineering departments at your site, the policy would probably be much looser.

The following policy violations from the sample policy filter *incom1.fil* generate level 3 alarms by default:

```
General tcp failure
General udp failure
ftp
telnet
smtp
DNS
gopher
finger
www
pop2
pop3
rpc
auth
nntp
ntp
exec
login
cmd
printer
ntalk
uucp
x11
udp dns
tftp
udp rpc
snmp
syslog
ntp
rip
icmp
tcp block
dns reply
last_block
tcp elite
tcp ftp data
tcp imap2
tcp imap3
udp netbios
tcp netbios ssn
tcp netstat
tcp snmptrap
tcp ssh fail
tcp
tcp at echo
tcp at nbp
tcp at rtmp
tcp at zis
tcp irc
tcp kerberos adm
tcp klogin
tcp kshell
tcp nfs
tcp systat
tcp tcpmux
tcp uucp rlogin
udp
udp at echo
udp at nbp
udp at rtmp
udp at zis
udp irc
udp kerberos adm
udp klogin
udp kshell
udp nfs
udp printer
udp snmp
udp systat
udp tcpmux
udp uucp
udp uucp rlogin
```

The following policy violation from the sample policy filter *incom1.fil* generates level 4 alarms by default:

```
large icmp fail
```

The following policy violations from the sample policy filters *incom1.fil* generate level 5 alarms by default:

```
IP spoofing
IP source routing
TCP frag header attack
TCP small frag attack
shun
```

DBMS REQUIREMENTS AND SETUP



In order for nr.sapd to stage data, you will need access to a database management system (DBMS). Oracle is the default database for the SAP package. Data can also be exported to Remedy's ARS system. If you do not have Oracle or Remedy ARS, you can customize *sapd* to support non-Oracle databases, such as Sybase and Informix.

Oracle Install and Setup

Oracle activation for nr.sapd consists of the following steps:

1. **Plan your Oracle environment.**
2. **Install/Mount the Oracle product.**
3. **Set up environment variables.**
4. **Verify generic connectivity.**
5. **Create user.**
6. **Verify NetRanger–Oracle connectivity.**
7. **Create tables.**
8. **Setup for Remedy ARS (Optional).**

Plan your Oracle Environment

Ask yourself the following questions about your environment:

- Do you have a local or remote database?
- How much disk space is allocated to the database?
- What tablespaces will be used for NetRanger data?

The answers to these questions will allow you to better install/mount Oracle.

Install/Mount the Oracle Product

You have two setup options for the Oracle Product: local or remote. WheelGroup strongly recommends that you export event data to a remote database. Staging data to a database running locally on the Director system could compromise existing monitoring functionality.

Whether you use a local or remote database, you will need to enter passwords for the Oracle users sys and system as part of the install process. You will use these passwords when setting up the Oracle NetRanger user.

NOTE

You will probably have to increase shared memory and semaphore parameters in /etc/system.

Local

Use the Oracle product CD-ROM and install with `orainst`. You will need Oracle Server (RDBMS) and SQL*Plus. Make sure to note the directory location of the install because you will use this in setting `$ORACLE_HOME`. Also note the SID, because you will use this to set `$ORACLE_SID`.

Remote

With the client/server model, Oracle RDBMS runs on a server and the client obtains access to it through proprietary DBMS network facilities, such as Oracle's SQL*NET. If you choose a remote setup for Oracle, you will need to choose one of the following setups:

- Remote homogeneous
- Remote heterogeneous

Remote homogeneous

If your client is the same operating system as the server, you can share the Oracle directory from the server and mount it to the client using normal NFS procedures.

Remote heterogeneous

If your client has a different operating system than the server, use the Oracle Installer with the product CD-ROM to install SQL*Plus and the TCP/IP Protocol adapter. Note the directory location of the install, because you will use this when setting `$ORACLE_HOME`. When the product binaries are available, you will need to configure $ORACLE_HOME/network/admin/tnsnames.ora.

The tnsnames.ora file maps a label (such as remoteDB) to the access protocols required to establish a connection to that database instance. A sample tnsnames.ora follows:

```
remoteDB =
(DESCRIPTION=
(LOCAL=NO)
(ADDRESS=
(PROTOCOL=TCP)
(HOST=DBserver)
(PORT=1521))
(CONNECT_DATA=(SID=nr)))
```

Where `remoteDB =` is the label used in the logon string, i.e., `username/password@remoteDB`. remoteDB is then expanded by SQL*NET into its underlying components:

- HOST=Dbserver
- Port=1521
- Protocol=TCP
- SID=nr

Ensure that the tnsnames.ora file is correctly located and configured; otherwise, you will get an Oracle TNS error message. Oracle will look for it in the following filepaths:

```
/var/opt/oracle/tnsnames.ora
$ORACLE_HOME/network/admin/tnsnames.ora
$HOME/.tnsnames.ora
```

UNIX Setup of Oracle Environment Variables

Oracle applications require a number of different environment variables whose settings depend on the setup you have chosen. Refer to the appropriate section for setting environment variables.

Remote or Local

Whether you are using a remote or local setup, Oracle requires the following three basic environment variables under UNIX: `ORACLE_HOME`, `PATH`, and `LD_LIBRARY_PATH`.

The following environment variables should be added to the .profile for the netrangr user account, and must contain the following information:

- $ORACLE_HOME is the base path for the Oracle product, such as /opt/app/oracle/product/7.1.6 or /usr/apps/oracle/app/product/7.3.2. The path usually ends in the version number of your Oracle product.
- PATH must contain $ORACLE_HOME/bin to allow access to the Oracle executables.
- LD_LIBRARY_PATH must contain $ORACLE_HOME/lib to allow access to the runtime libraries.

Local Only

If using a local setup, Oracle requires another environment variable, ORACLE_SID, which specifies what data area to use with a local database. When this is the case, update ORACLE_SID in the .profile as above.

tty problems (HP-UX)

On certain platforms, such as HP-UX, the default tty settings use the character @ for some built-in commands. You will have a problem connecting to a remote database because the tty settings will delete a portion of your connect string username/password@remoteDB.

To examine the tty setting, execute the following command:

```
% stty -a
```

This will show the mappings for commands. If one of the commands, such as kill, is mapped to @, you will need to change it by executing the following command:

```
% stty kill ^Z
```

Verify Generic Connectivity

Verify that Oracle access is enabled.

Before attempting an Oracle connection, ensure that $ORACLE_HOME, $PATH, and $LD_LIBRARY_PATH are set correctly (as described above).

Local

Set `$ORACLE_SID`. The Oracle daemons must be running. Use the Oracle executable `sqlplus` to verify basic connectivity to Oracle with the following command:

```
% sqlplus sys/password
```

If successful, you will enter into interactive SQL. You will see a prompt similar to the following:

```
Connected to:
Oracle7 Server Release 7.3.2.1.0 - Production Release
PL/SQL Release 2.3.2.0.0 - Production
SQL>
```

NOTE

Ideally, you should omit the password string in the command line call. `sqlplus` will then prompt you for the password. This prevents the password from showing up in output from system applications such as `ps`.

Remote

Use the Oracle utility `tnsping` to verify that the basic network setup is correct (this is similar to the standard ping utility) with the following command:

```
% tnsping remoteDB
```

Use `sqlplus` to verify remote connectivity by appending a valid remoteDB string onto the end of the connection string:

```
% sqlplus sys/password@remoteDB
```

Refer to Appendix A: TROUBLESHOOTING for information about the common error messages you might encounter.

Create an Oracle User

Follow these steps before you create a user:

1. **Log into Oracle as sys or system (using `sqlplus` or `sqldba`).**
2. **Determine what Oracle data areas (tablespace) will be assigned for NetRanger usage.**

NOTE

This tablespace needs to be as large as possible. The option of assigning a default tablespace to a user eliminates the problem of having to direct subsequent commands to use a certain tablespace. For high volume NSX sensors, you may need to dedicate a TEMPORARY TABLESPACE and a larger ROLLBACK area to the NetRanger user.

You are now ready to create a user. Follow these steps to create a user:

1. **Create the user with the following command:**

```
SQL>  create user NetRanger identified by PASSWORD default tablespace 'USERS'
```

Where `PASSWORD` is your secret password and `USERS` is your designated tablespace.

2. **Activate the account with the following command:**

```
SQL>  grant dba to NetRanger
```

(Or other lesser privilege based on your site's security policy; at a minimum, the NetRanger user will need privilege to CONNECT, SELECT, INSERT, and DELETE.)

NOTE

Make sure you update the sapd.conf tokens DBUser and DBPass with the new user and password. *sapd* should not be activated until these are set (unless you are customizing the scripts). Use nrConfigure to change these tokens.

Verify NetRanger–Oracle Connectivity

Verify that user netrangr can access the Oracle NetRanger account by running `sqlplus` from the Director system.

Create Tables

Once you have verified Oracle connectivity and the user netrangr, you can proceed with the creation of the NetRanger database tables. These database tables must be created before *sapd* can start loading the data. Begin by connecting to Oracle as user netrangr via `sqlplus` from the /usr/nr/bin/sap/sql/skel directory.

Once connected, execute the following command at the `SQL>` prompt:

```
SQL> @nrdb_master_create
```

NOTE

The first time you use this procedure, you will receive error messages that say that the tables do not exist. This is normal on the first install. Be careful when using these create scripts, because they will drop the existing table, thus deleting all the data.

Setup for Remedy ARS (Optional)

Use schema provided in /usr/nr/bin/sap/skel/nr.alarm.schema.def. Refer to Chapter 5 for more information about Remedy ARS.

Non-Oracle Install and Setup

Please refer to your DBMS software documentation for installation and setup. Use the *Oracle Install and Setup* section as a template.

DBMS Setup Checklist

The following checklist aids in setting up a DBMS. It also provides a high-level overview for Appendix E, in which can be found full details of what follows.

Install and Configure Oracle

_____ Install Oracle Product: CD-ROM or NFS mount

_____ Configure .profile: `ORACLE_HOME, PATH, LD_LIBRARY_PATH`

_____ Set up SQLNET: tnsnames.ora

_____ Verify SQLNET: sqlplus or tnsping

NetRanger–Oracle Setup

_____ Create user

_____ Verifty UNIX netrangr user login to Oracle as netranger

_____ Create tables

nr.sapd Setup

_____ Configure /usr/nr/etc/sapd.conf: set DBUser and DBPass

_____ Configure /usr/nr/etc/daemons: add nr.sapd to list

_____ Execute: `nrstart`

_____ Verify that *sapd* is running:

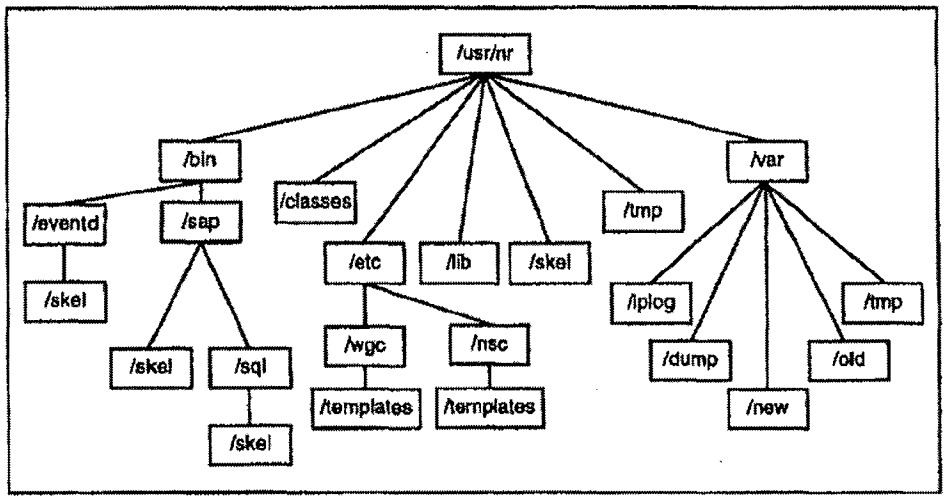
```
tail -f /usr/nr/var/messages.sapd
```

THE NSX FILE STRUCTURE



One of the quickest ways to gain a detailed understanding of NetRanger is to review the NSX file structure and underlying elements. A thorough knowledge of the NSX file structure will help you configure and install the NSX as well as help you better utilize NetRanger's features. It should be noted that this file structure serves as the foundation for both the Director and NSX systems.

The default directory location for the NSX subsystem is /usr/nr, which contains the following directory structure:



This appendix provides a brief explanation of each of these directories.

~/bin

This directory contains all of the executables required to run and administer the application services that support an NSX or Director system. These can be loosely grouped into three basic categories:

- Daemon Applications
- Configuration Commands
- System Commands

Daemon Applications

As discussed in Chapter 1, NetRanger is a collection of discrete subsystems that have been implemented via daemons, each with its own well-defined set of services.

NetRanger is also a highly configurable distributed application. The daemon services running on an NSX or Director host depends on the configuration of the application as a whole. By default, an NSX host must have *sensord*, *managed*, *loggerd*, and *postofficed* daemons running. A Director system should have *postofficed*, *smid*, *loggerd*, and *configd* running. In a multi-tiered configuration, one Director system could be configured to only stage data to an RDBMS via *loggerd* and *sapd*, while another Director system could be dedicated to monitoring and response via the *smid* and *configd* daemons.

The full complement of NSX daemons is as follows:

- nr.configd
- nr.eventd
- nr.loggerd
- nr.managed
- nr.postofficed
- nr.sapd
- nr.smid
- nr.sensord
- Nr.InfoServer

Each of these daemon services is briefly described on the following pages.

configd

This daemon reads and writes data to the NSX configuration files (which are currently restricted to *~/etc*). All communication with this daemon is controlled by the following set of SNMP-like executables:

- nrget & nrgetbulk
- nrset & nrunset
- nrexec

NOTE

This daemon provides an *interface* to configuration information. Actual reconfiguration is performed by *managed*. Both command line and graphical user interfaces are provided.

---

eventd

This daemon is designed to run on Director systems. It allows the Director system to generate user-defined actions for events received by *smid*. A common action is to generate pager notifications via e-mail for alarms of severity 4 and above.

loggerd

This daemon writes out sensor and error data to flat files generated by one or more of the other daemon services. This data is passed to *loggerd* via *postofficed*. *loggerd* creates two basic types of flat files: a single NSX Event file, and one or more IP Session logs. As noted in the *Overview*, data is written to flat files for reasons of performance and fault tolerance. This data can be staged into an RDBMS and managed for archival via *sapd*.

managed

Whereas *sensord* is responsible for interpreting packet filter events, *managed* is responsible for managing and monitoring these devices. For example, when *sensord* identifies that a certain type of attack should be shunned, it sends a shun command to *managed* via the post office facility. *managed* then reconfigures the packet filter appropriately. Similar commands can be sent to this daemon from the Director. *managed* can also be polled for operational statistics such as:

- The number of network devices being managed.
- The device type.
- Device statistics (packets/sec, bytes/sec, etc.).

postofficed

This daemon serves as the communication vehicle for the entire NetRanger product. All communication between daemons is routed through this service. In most NetRanger configurations the NSX and Director systems reside on separate hosts. Each system relies on a *postofficed* to maintain communication with the other. Note that a *postofficed* is required even when all of the systems are located on a single host.

This daemon service include the following features:

- A proprietary connection-based protocol that resends a message whenever any of its packets are lost.
- Point-to-point routes that may include intermediate post office nodes.
- Communication integrity that is maintained via alternate routes (specified in ~/etc/routes and ~/etc/destinations). When one route fails, communication is switched to an alternate route. Communication is reestablished to the preferred route as soon as it comes back online.

Routing is based on a three-part key that includes the following tuples:

- Organization
- Host
- Application

The Organization IDs are defined in ~/etc/organizations; Host IDs are defined in ~/etc/hosts; Application IDs are defined in ~/etc/services.

sapd

This daemon is a user-configurable scheduler that controls database loading and archival of old event and IP session logs. Oracle or Remedy database loading is accomplished with *sapx*. Data archival is controlled with the native SAP dump scripts.

smid

This daemon routes messages to and from the Director and other daemon services, such as *sensord*. This service relies on routing information stored in the Director's ~/etc/hosts file. In addition to being able to communicate with other daemons, *smid* can redirect messages to other daemon services, such as *eventd* and *loggerd* based on *DupDestination* entries in etc/smid.conf.

sensord

This daemon interprets and responds to all of the events generated by one or more packet filter devices. Data is read in from Network Device sockets and sent to *postofficed* for routing to other daemon services. The types of events received by *sensord* is a function of the filter policy on each packet filter device. The types of messages *sensord* sends to other daemons is based on information stored in ~/etc/sensord.conf. Where these messages are routed is defined in ~/etc/destinations.

Although *sensord* supports 256 levels of alarm events (0-255), only levels 0-6 are currently being used. Level 0 is an internal alarm that *sensord* uses to track such services as RIP, ICMP, and so on; it is never routed to *postofficed*. Levels 1-6 are sent to *postofficed*, and they identify increasing patterns of misuse, where 5 is the highest type of alarm. Note that every alarm message includes an Alarm-ID that identifies a specific attack signature. These alarm signatures are enumerated in ~/etc/sensord.conf as SigOfGeneral entries.

In addition to generating alarm messages for a Director application, *sensord* can automatically instruct *managed* to shun an attack for a specified period of time.

Configuration Commands

The NSX Sensor system comes with the following built-in commands that allow remote devices to configure and gather information about the NSX. As noted in *Operating the Director*, these commands serve as the foundation for the Director's **nrConfigure** interface.

- nrget
- nrgetbulk
- nrset
- nrunset
- nrexec

nrget

This file retrieves single pieces of information from the NSX.

```
nrget socket appid hostid orgid priority token [ identifier... ]
```

nrgetbulk

This file retrieves multiple pieces of information from the NSX.

```
nrgetbulk socket appid hostid orgid priority token [ identifier... ]
```

nrset

This file sets attributes about the NSX.

```
nrset socket appid hostid orgid priority token [identifier...] value [value ...]
```

nrunset

This file unsets attributes about the NSX.

```
nrunset socket appid hostid orgid priority token [identifier...] value [value... ]
```

nrexec

This file executes commands on the NSX.

```
nrexec socket appid hostid orgid priority timeout token [identifier...] value
[value ... ]
```

System Commands

The following scripts are used to perform maintenance on the NSX:

- install
- nrstart
- nrstop
- nrstatus

install

This script is used to install the startup/shutdown files for NetRanger into the UNIX /etc/initd directory. The command takes the following form:

```
install ( add | remove )
```

nrstart

This script starts the NetRanger daemons. It supports the following options:

- -d Don't daemonize
- -h Show help
- -v Show version number

The command takes the following form:

```
nrstart [-d] [-h] [-v]
```

nrstop

Executing this script stops the NetRanger daemons.

nrstatus

Executing this script returns the currently running daemons.

~/classes

This directory serves as the *rooted class path* for the Director's Java applications, such as **nrConfigure.**

~/etc

This directory contains two types of configuration files: **daemon-specific** and **NSX system** files. All of NetRanger's configuration files are currently ASCII flat files.

Daemon Configuration Files

There is a one-to-one correspondence between daemons and their configuration files. Whereas the naming convention for a daemon is `nr.<daemon>`, the convention for its configuration file is `<daemon>.conf`. Each of these files contains token-based configuration information of the form: `<token> <value>`. For example, the error file for *nr.managed* is identified in *managed.conf* as follows:

```
FilenameOfError../var/errors.managed
```

Although these entries can be set via a text editor, all changes should be managed via **nrConfigure.** Otherwise, the chance of generating errors increases.

- configd.conf
- eventd.conf
- loggerd.conf
- managed.conf
- postofficed.conf
- sapd.conf
- sensord.conf
- smid.conf

With the exception of *managed.conf*, *sensord.conf*, and *smid.conf* these files currently contain little more than timing intervals and the name of the ~/var/error file for each daemon service. The remainder of this section is devoted to explanations of *managed*, *sensord*, and *smid.conf*.

managed.conf

This file is used by the NetRanger NSX to configure the management daemon that controls actions on the router. Two types of tokens in this file require special attention: *NetDevice* and *NeverShunAddress.*

NetDevice identifies the **IP address** and the **system password** of an NSG BorderGuard the NSX Sensor is connected to. A separate *NetDevice* entry is required for every BorderGuard system being managed by the NSX system. The format of this token is as follows:

```
NetDevice ROUTER_IP_ADDRESS NSG_Borderguard_v1  ROUTER_PASSWORD
```

NeverShunAddress identifies the IP address of a host or network device you never want the NSX or BorderGuard to shun. At the very minimum you make sure that this file contains *NeverShunAddress* entries for the local system and the remote NSX or Director system. Other addresses can be added based on your operational needs. Example entries might look as follows, where <255.255.255.255> identifies the subnet mask for the IP address you do not want shunned.

```
NeverShunAddress ROUTER_IP_ADDRESS      255.255.255.255

NeverShunAddress NSX_IP_ADDRESS         255.255.255.255

NeverShunAddress INTERNAL_NETWORK       255.255.255.255
```

NOTE

NeverShunAddress entries do not bypass existing filters, but rather prevent them from being completely blocked out of the router.

sensord.conf

This file serves as the heart of the NSX Sensor system, and is the largest of all of the configuration files. It can be logically broken down into the following sections:

- General
- Event Specification
- Strings To Look For
- Matched String and Event Levels
- TCP/UDP Ports and Event Levels
- Policy Violations
- Excluded Events

General

At a minimum, you will need to edit the following lines:

```
RecordOfInternalAddress INTERNAL_NETWORK_ADDRESS INTERNAL_NETWORK_NETMASK

RecordOfInternalAddress  ROUTER_EXTERNAL_IP_ADDR   255.255.255.255
```

These two lines establish which networks NetRanger will protect. **It is important to include all internal addresses because this information is also used for intrusion detection.** You should have one entry for each network address and one for the router external interfaces.

```
RecordOfDataSource      ROUTER_IP_ADDRESS   255.255.255.255
```

This line determines which BorderGuard the sensor daemon will receive information from. You need to fill in the router IP address for this line.

```
RecordOfLogAddress   LOGGED_NETWORK_ADDRESS   255.255.255.0
```

This line establishes a binary log of a particular host or network. This is most commonly used to support an investigation and creates a binary record of all data originating from a particular source address. Initially, this line will probably be commented out.

```
MinutesOfAutoLog       LOG_MINUTES
MinutesOfAutoShun      SHUN_MINUTES
```

These two lines establish the amount of time that NetRanger will shun or log after a particular alarm is received.

Event Specification

The first task-specific segment is Event Specification, which is based upon the SigOfGeneral token. This token identifies an *action* and *alarm levels* for various types of events. *sensord.conf* contains a number of different `SigOf` tokens, all of which share the same basic format, which is defined as follows:

```
SigOfGeneral  SIGID   ACT   D1   D2   D3   D4   # Description of event
```

NOTE

`SIGID` is a numeric identifier that establishes a logical link to internal signature IDs in the *sensord* daemon, and should be treated as **read-only** information. Any changes or additions to these entries will disrupt NSX functionality.

The first column that can be configured is the `ACT` column. It contains a number that specifies how to respond (an *action*) to a particular event. Valid `ACT` settings include:

```
0 -  no action
1 -  shun
2 -  log
3 -  shun and log
```

The remainder of SigOfGeneral maps alarm levels to each of the destinations specified in /usr/nr/etc/destinations. This makes it possible to uniquely define the severity of an event for every destination the NSX is configured to talk to.

The following SigOfGeneral defines what to do when a `TCP Port Sweep` is encountered. It first states that no automatic action should be taken (`ACT=0`). It also defines that level 5 alarms should be sent to its first three destinations, and a level 4 alarm to the last destination.

```
SigOfGeneral  3001   0   5   5   5   4   # TCP port sweep
```

Strings To Look For

This section defines the strings to look for within a given TCP/IP service. The general format for an entry in this section is the following:

```
RecordOfStringName  StringID  TCP/IP Port  Direction  Num Occurrences  "Matched String"
```

StringID establishes the unique ID for that particular string. TCP/IP port tells NetRanger which port to watch to look for this string. Note that *first.fil* must have a Copy_To entry for this port that copies all of its packets to the NetRanger NSX. Direction tells the NSX which direction to look for this string:

```
1 - external-to-internal
2 - internal-to-external
3 - both directions
```

The number of occurrences indicates how many times the NSX must see this string before it alarms. Matched String is the actual string to be matched. The exact format is controlled by regular expression (regex) syntax.

Matched String and Event Levels

This section identifies action and alarm levels for the string matches defined in the previous section. SigOfStringMatch uses the same format as SigOfGeneral described above, and is linked to a RecordOfStringName by its StringID.

```
SigOfStringMatch StringID  ACT  D1  D2  D3  D4  # Description of event
```

TCP/UDP Ports and Event Levels

This section establishes the alarm levels for any packet, successful or not, with a TCP or UDP destination port corresponding to the entry. The action fields and the event level specification are the same as in previous sections.

The following entry illustrates that all Telnet attempts (Port 23) should have no action taken but should generate a level 2 alarm:

```
<port> <action> <dest1> <dest2> <dest3> <dest4>

SigOfTcpPacket    23      0      2      2      2      2
```

The next entry indicates level 3 alarms for all TFTP (port 69) attempts which logs the packets:

```
<port> <action> <dest1> <dest2> <dest3> <dest4>

SigOfUdpPacket    69      1      3      3      3      3
```

Policy Violations

This section identifies the filter names and IDs for all BorderGuard filters that can pass information to the NSX. In the following example, Income_1_Filter_Fail has a filter ID of 1000.

```
RecordOfFilterName  1000   Income_1_Filter_Fail
```

Each of these policy filters has a matching SigOfFilterName that identifies its action and destinations. The format is identical to the SigOf- tokens described above. In the following example an `Incom_1_Filter_Fail` event will be logged to the second destination defined in /usr/nr/etc/destinations.

```
<filter_id> <action> <dest1> <dest2> <dest3> <dest4>

SigOfFilterName   1000      2      2      2      2      2
```

Excluded Events

The final section of */usr/nr/etc/sensord.conf* is the Excluded Events segment. This is where alarms can be *locked out* for events at specific source addresses. The standard format follows:

```
RecordOfExcludedAddress   SigID   SubSigID     IP Address
```

`SigID` corresponds to the primary signature established in the Event Specification section that establishes all of the primary IDs. `SubSigID` corresponds to those signatures established in the latter sections, such as those for filtering and string matching.

smid.conf

This file is the configuration file for *smid,* which passes information to the Director *nrdirmap* application for display within the OpenView user interface. The only token that needs to be edited in this file is `DupDestination`, which defines duplicate destinations for the events. This token allows a Director system to forward event information onto other services and systems. In the following example, events, commands, and errors of level 1 and above are forwarded onto *loggerd.* This allows the Director to monitor ***and*** log events based on a single NSX data stream. This in turn, helps reduce the amount of network traffic an NSX must transmit to a Director. Instead of having to transmit two identical (but separate) data streams to `director1.wheelgroup`, the NSX is able to send a single stream that is duplicated on the Director platform.

```
DupDestination   director1.wheelgroup   loggerd   1   ERRORS,COMMANDS,EVENTS
```

As the next example shows, this token can also be used to forward events onto another Director system. In this case, however, only events of level 2 and above are forwarded.

```
DupDestination   director2.wheelgroup   smid   2   ERRORS,COMMANDS,EVENTS,IPLOGS
```

NOTE

The host and organization IDs must match entries defined in /usr/nr/etc/hosts.

NSX System Files

The */usr/nr/etc* directory contains all of the NSX's system files. These files are modeled after Unix */etc* files and Unix system administrators should have no problem understanding their format and entity relationships. The files currently are as follows:

- organizations
- hosts
- services
- auths
- destinations
- routes
- daemons
- signatures

NOTE

These files are *described* in order of use rather than alphabetically.

organizations

This file identifies all of the *organizations* and their *organization ids* currently registered for a given NetRanger system. The format of the file is:

```
<org_id><organization_name>
```

A typical file might look as follows:

```
100     companyabc
101     companydef
102     companyghi
1001    companyjkl
```

NOTE

This file serves as a *global* registry that must agree across all of the NSX and Director systems within a NetRanger domain.

hosts

This file is much like a UNIX /etc/hosts file. It enumerates the organizations and hosts that are recognized by a given NSX or Director system in a NetRanger configuration. Each entry has the following format:

```
<host_id>.<organization_id><host_name>
```

where `host_id` and `organization_id` are numeric identifiers assigned by WGC, and *`host_name`* is a DNS-like name assigned by the user. In addition to a list of recognized hosts, this file always contains a *localhost* entry.

A typical hosts file might look like the following:

```
10.100    localhost
10.100    admin.companyabc
11.100    dev.companyabc
12.100    data.companyabc
13.100    nsx-1.companyabc
14.100    nsx-2.companyabc
15.100    nsx-3.companyabc
```

NOTE

Entries for each host must be consistent across *all* NSX and Director systems. Otherwise, some systems will generate "unrecognizable host" messages in ~/var/errors.postofficed.

services

This file defines the *application_id* for each of the daemon services found in **~/bin**. When combined with the *host_id* and *organization_id*, these identifiers uniquely identify a daemon within a NetRanger security map. Each entry has this form:

```
<application id><daemon name>  [<comment>]
```

All NSX hosts should have the same default services file, which is currently defined as follows:

```
10000   postofficed     # Provides the NetRanger comm system
10001   sensord         # Monitors network traffic for security
10002   configd         # Sets and Gets configuration information
10003   managed         # Manages NetRanger components
10004   eventd          # Sends messages and alarms to pagers
10005   loggerd         # Logs events, cmds, errors, IP logs, etc.
10006   smid            # Routes msgs & events to nrdirmap
10007   sapd            # Stages log data into a DBMS
```

auths

This file identifies which type of *configd* operations are authorized for the each of the hosts listed in this file. Each entry follows this form:

```
<host name>        <configd services>
```

In the following example, `admin.wheelgroup` has full authorization, whereas `data.wheelgroup` can only read configuration information.

```
admin.wheelgroup   GET,GETBULK,SET,UNSET,EXEC
nsx-1.wheelgroup   GET,GETBULK
nsx-2.wheelgroup   GET,SET,UNSET,EXEC
```

NOTE

Each entry must have a corresponding entry in ~/etc/hosts.

destinations

NetRanger is a distributed application that has the ability to route messages from a given post office to any of the daemon services registered in the ~/etc/hosts and ~/etc/services files. The destinations file identifies what type of messages get routed to a given application on a given host. Each entry is of the following form:

```
<Destination Id.> <host name> <daemon name> <message level>  <message type>
```

where <Destination Id> is a Numeric ID (up to 32 destinations are currently allowed); <host name> must be an entry from ~/etc/hosts; <daemon name> must be an entry from ~/etc/services; <message level> identifies the level of messages that will be routed (messages below that level are not passed on to *postofficed*); <message type> identifies the types of messages to route.

For example, in the following destinations file, level 2 (and higher) messages of the type *event, error, command,* and *IP_log* are being routed to the *smid* daemon on the host *admin.wheelgroup*. This type of configuration would normally be found on an NSX system, where *admin.wheelgroup* is a Director system.

```
1   admin.wheelgroup   smid   2 EVENTS,ERRORS,COMMANDS,IPLOGS
```

routes

This file identifies the actual IP routes that *postofficed* uses to send messages between different hosts. Each entry is of the following form:

```
<host name>  <connection #>  <IP address>  <UDP port>  <Type>
```

<host name> must be an entry from ~/etc/hosts. <connection #> identifies the priority of this entry relative to the other routes enumerated in this file. The lower the number, the higher the route priority. If there is more than one entry of the same priority, *postofficed* accepts the last entry of that priority. <IP address> identifies the **end-point** IP address of the **remote system.** <UDP port> identifies the UDP port service to route through on each host. These ports are usually in the 30-50000 range. <Type> identifies whether the connection is a dial-up vs. dedicated connection. This field is currently not used.

In the following example, nsx-2.wheelgroup serves as the primary route to the IP Address 10.1.7.12, and nsx-3.wheelgroup is the secondary route to the same host.

```
nsx-1.wheelgroup      1      10.1.7.9       45000      1
nsx-2.wheelgroup      1      10.1.7.12      45000      1
nsx-3.wheelgroup      2      10.1.7.12      45000      1
```

daemons

This file contains the names of daemons that should be started when the system starts up. Each entry is of the following form:

```
<daemon name>
```

For example, the following sample entry would start *postofficed*, *configd*, *loggerd*, and *smid* on a Director system:

```
nr.postofficed
nr.configd
nr.loggerd
nr.smid
```

NOTE

Whereas the *services* file identifies all of the daemon services a given NSX or Director system is *capable* of running, the *daemons* file identifies the services to launch upon startup.

signatures

This file associates *signature ids* with *signature names* for NetRanger applications, and is used primarily by Director systems. The format of each entry is in the file is:

```
<signature id> "<signature name>"
```

For example, the following sample entry defines a few attack signatures.

```
1000 "Bad Option List"
1001 "Record Packet Rte"
1002 "Timestamp"
1003 "Provide s,c,h,tcc"
1004 "Loose Src Rte"
1005 "SATNET ID"
```

NOTE

This file should NOT be modified.

~/skel

This directory contains basic UNIX configuration files (such as .profile) that are required to configure remote login accounts.

~/tmp

This is the directory where sockets are created by the different daemons. The names of these sockets are dictated by the parent daemon. *postofficed* creates sockets with names such as mailbox.sensord and mailbox.configd. *configd* creates sockets with names such as socket.command.

~/var

This is the default location for all of the log files generated by *loggerd* and error files for all of the applications listed in the ~/etc/daemons file.

Log Files

Two different types of log files are created by *loggerd:* **Event** and **IP Session** logs.

Event File

Although the name of this file can be changed via nrset, the default name of this file is currently hard-coded in *loggerd.* The name of event logs are based on the date and time a new log began (e.g., log.199607291332).

Message Type	Description
0	Default
1	Command
2	Error (Defined)
3	Command Log (Defined)
4	Event (Defined)
5	IP Log
6	Redirect

The schema for defined message types are as follows:

Error records from *managed* (ERROR)

```
1, MsgType (SInt) nrPROTOCOL_ERROR,
2, ULong RecordID,
3,4, GMT timestamp (String year/month/day, hour:min:sec)
5,6 Local timestamp (String year/month/day, hour:min:sec)
6, ULong addr.ApplID,
7, ULong addr.HostID,
8, ULong addr.OrgID,
9, String ErrorData
```

Command Log records from *configd* (COMMANDLOG)

```
1, MsgType (SInt) nrPROTOCOL_COMMANDLOG,
2, ULong RecordID,
3,4, GMT timestamp (String year/month/day, hour:min:sec)
5,6 Local timestamp (String year/month/day, hour:min:sec)
6, ULong addr.ApplID,
7, ULong addr.HostID,
8, ULong addr.OrgID,
9, ULong src.ApplID,
10, ULong src.HostID,
11, ULong src.OrgID,
12, String LogData
```

Event/Alarm records from *sensord* (PROTOCOL_EVENT)

```
1, MsgType (SInt) nrPROTOCOL_EVENT,
2, ULong RecordID,
3,4, GMT timestamp (String year/month/day, hour:min:sec)
5,6 Local timestamp (String year/month/day, hour:min:sec)
6, ULong addr.ApplID,
7, ULong addr.HostID,
8, ULong addr.OrgID,
9, String from,
10, String to,
11, SInt event->level,
12, ULong event->SigID,
13, ULong event->SubSigID,
14, String protocol,
15, String sourceIpAddress,
16, String destIpAddress,
17, SInt event->SrcPort,
18, SInt event->DstPort,
19, String routerIpAddress,
20, String EventData
```

Session Log Files

In addition to detecting attack signatures, *sensord* is able to monitor the UDP traffic associated with a specific type of attack. For example, *sensord* can be configured to monitor all of the packets associated with an IP spoof. *loggerd* creates a separate log file in ~/var/iplog for each of these monitoring sessions. The name of each session log file is based on the IP address of the attacking host (e.g., ip.log.209.15.163.54).

NETRANGER HARDWARE COMPONENTS



NSX Sensor Hardware Components

- Rack-Mounted PC w/Solaris x86 2.5
- 166 MHz Pentium© processor (10Mbps)
- 110 MHz UltraSPARC© (100Mbps)
  - wide SCSI
- 32 MB RAM
- Three NSX Versions:
  - NSX 1000
  - NSX 2000
  - NSX 5000
- Four configuration options:
  - access through serial port
  - network access
  - modem dial-up
  - video/keyboard

NSX Versions

NSX 1000

- Pentium 166 NSX
- NSG BorderGuard 1000
  - 2 10BaseT/AUI Ethernet
  - 3 WAN Daughtercards/1 Ethernet
  - ISDN
- Up to 512 Kbps



Figure F.1:The NSX 1000

NSX 2000

- Pentium 166 NSX
- NSG BorderGuard 2000
  - 4 Ethernet
  - 4 WAN
  - 2 Ethernet/2 WAN
- 512 Kbps to 10 Mbps Ethernet



Figure F.2:The NSX 2000

NSX 5000

- UltraSPARC NSX
- NSG ERS/Passport
  - FDDI
  - Ethernet
  - WAN (T1 and T3)
  - (ATM)
- 10 to 100 Mbps Ethernet



Figure F.3:The NSX 5000

NSG BorderGuard Hardware Components

- The BorderGuard is a bridge/router that links IEEE 802.3/Ethernet networks, and other networks via WAN links (e.g., ISDN, leased lines).
- Includes Packet Control Facility (PCF) for firewall routing
- Includes Bridge Control Facility (BCF) for filtered bridging
- Routing Support
  - TCP/IP
  - DECnet Phase IV
  - Novell IPX
  - Banyan VINES
  - AppleTalk II
  - XNS

UNINSTALLING NETRANGER



The following procedure describes how to uninstall NetRanger software.

NOTE

This procedure does not describe how to remove HP OpenView from Director systems. **If you want to remove OpenView,** uninstall the NetRanger software first, then refer to the HP OpenView documentation for the OpenView removal procedures.

If you want to remove the NetRanger Director software but continue to use HP OpenView, you should delete your NetRanger Director data from the OpenView databases before uninstalling the Director. Once you uninstall the Director, there will be no way to remove NetRanger data from the databases, other than by completely removing the databases. To delete data from the databases while the Director is still installed, use the Delete Object menu item from the OpenView user interface.

1. **Login as user** `root`.
2. **Copy the NetRanger software removal utility to the /tmp directory by typing:**

```
cp /usr/nr/bin/nrUninstall /tmp
```

3. **Run the NetRanger software utility by typing:**

```
/tmp/nrUninstall
```

4. **Pick the system that you want to remove (usually, this will be option number 1 - All NetRanger Packages).**
5. **A list of packages to be removed will be displayed. Press** `y` **to continue.**
6. **For each package, status of the removal process will appear on the screen. Read the information and then press return to continue.**

7. **If you have chosen to remove all of the NetRanger packages, then you will be asked if you want to remove the `netrangr` user and group.**

NOTE

Before you choose to remove the netrangr user, make sure that all netrangr users have logged off first.

---

8. **Press y to remove the netrangr user and group or n to preserve them.**

NETRANGER *managed* FAQ



This appendix lists the Frequently Asked Questions (FAQ) for *managed*, the process that the NetRanger system uses to manage the NSX, the Director, and various network devices.

Where is *managed*?

nr.managed, located in /usr/nr/bin, is usually run on the NSX to support network devices such as the BorderGuard and Passport.

How do I configure *managed*?

managed uses its default configuration file, managed.conf, located in /usr/nr/etc, for initialization upon startup. Like all NetRanger daemons, this ASCII file can be modified and *managed* will reprocess it upon a HUP signal. Configuration tokens can also be used to adjust the configuration of *managed* on the fly. The default configuration file, the error reporting file, and other configuration items may be changed via tokens.

What is a network device?

The term network device is used because *managed* supports multiple network devices. The current list of supported devices includes the following:

Hardware Device	Operating Systems	Operational Mode
NSG BorderGuard 1000	3.0 & 4.0	Routing & Bridging (v 4.0 only)
NSG BorderGuard 2000	3.0 & 4.0	Routing & Bridging (v 4.0 only)
NSG Passport	Various	Routing & Bridging

What does a network device do?

A network device must sit within the flow of traffic within a network. This is accomplished by the network device acting as either a router or a bridge. A facility must support the selective copying of certain packets (TCP/IP, IPX/SPX, etc.) to the NSX for analysis. The device must also support the capability to be dynamically reconfigured to enforce a security policy. The BorderGuard and Passport provide this through the use of NSG's NetSentry.

How can I determine how many network devices are currently configured?

Using the token, NumberOfNetDevices, you can perform a **get** operation to return the number of network devices currently configured within *managed*. The following example shows how the *managed* queried is currently working with 4 network devices.

```
netrangr% nrget 10003 10 100 1 NumberOfNetDevices
4
```

How many network devices does *managed* support?

There is no predefined limit to the number of devices supported by *managed* other than that imposed by performance constraints. Up to a dozen network devices can be easily handled with little performance impact to the NSX. Hundreds of network devices may be configured, although WheelGroup has not tested this and does not guarantee that this will work.

How do I configure *managed* to work with a network device?

There are two ways of configuring *managed* to work with a network device: through the configuration file and through **GET**, **GETBULK**, **SET**, and **UNSET** commands. The token used to configure a network device is NetDevice. In the configuration file, the format is the following:

```
NetDevice <IP Address> <Type> <Password>
```

These fields specify the network IP address, the type of network device, and the password that NetRanger should use. If a network device supports user accounts, like the Passport, *managed* currently expects that user account to be for user netrangr. BorderGuards do not support user accounts. Although *managed* will be able to support this successfully if password protection is disabled, WheelGroup recommends against this practice. The type field is used for backward compatibility with the configuration files in NetRanger versions 1.0 and 1.1. *managed* no longer uses this information because it is capable of determining the type of network device by itself. However, the type field MUST BE present or the password field will not be properly accessed. Therefore, the string **foo** would be adequate for the type description. Here are some examples:

```
NetDevice  10.1.2.1  NSG_BorderGuard  3j8c7Hke
NetDevice  10.1.6.90 NSG_BorderGuard  pdk6Y5nr
NetDevice  10.1.7.2  NSG_Passport     wG7ki8En
```

Network devices may be added/removed on the fly through **SET** and **UNSET** operations. The following example uses the commands **nrset** and **nrunset** to dynamically add and remove devices on an NSX with a hostid of 10 and orgid of 100. The priority of the requests is 1. The NetRanger Director's NSX configuration tool also supports these operations.

```
netrangr% nrset 10003 10 100 1 NetDevice 10.1.4.1 NSG_BorderGuard 3j8c7Hke
Success

netrangr% nrunset 10003 10 100 1 NetDevice 10.1.2.1
Success
```

How can I tell which network devices are currently configured?

Using the NetDevice token, **get** and **getbulk** operations may be performed to identify which devices are in operation. Here are a few examples using the same NSX as discussed above:

```
netrangr% nrgetbulk 10003 10 100 1 NetDevice
10.1.2.1  NSG_BorderGuard_2000  3j8c7Hke
10.1.6.90 NSG_BorderGuard_2000  pdk6Y5nr
10.1.7.2  NSG_Passport      wG7ki8En

netrangr% nrget 10003 10 100 1 NetDevice 10.1.2.1
NSG_BorderGuard_2000 3j8c7Hke
```

How can I determine what the hardware type is for a network device?

Use the NetDeviceType token to perform **get** and **getbulk** operations to identify what type of hardware a network device is. Here are a few examples:

```
netrangr% nrgetbulk 10003 10 100 1 NetDeviceType
10.1.6.90 NSG_BorderGuard_2000
10.1.2.1 NSG_BorderGuard_2000
10.1.6.2 NSG_Passport

netrangr% nrget 10003 10 100 1 NetDeviceType 10.1.2.1
NSG_BorderGuard_2000
```

How can I determine the operating system version of a network device?

Use the NetDeviceVersion token to perform **get** and **getbulk** operations to identify the operating system version for a network device. Here are a few examples:

```
netrangr% nrgetbulk 10003 10 100 1 NetDeviceVersion
10.1.6.90 4.0
10.1.2.1 4.0
10.1.6.2 BC01S1D

netrangr% nrget 10003 10 100 1 NetDeviceVersion 10.1.2.1
4.0
```

How can I tell if *managed* is properly working with a network device?

Use the NetDeviceStatus token to perform **get** and **getbulk** operations to identify the current status of the communication link between *managed* and the network device. Here are a few examples:

```
netrangr% nrgetbulk 10003 10 100 1 NetDeviceStatus
10.1.6.90 Active
10.1.2.1 Connecting
10.1.6.2 Configuring

netrangr% nrget 10003 10 100 1 NetDeviceStatus 10.1.2.1
Active
```

The valid values of the connection status are as follows:

Status	Description
Inactive	The link is completely down*
Connecting	*managed* is trying to connect via a Telnet connection
Login_Sent	*managed* has sent the username netrangr to the login prompt
Password_Sent	*managed* has sent the password to the password prompt
Configuring	*managed* is configuring the network device
Active	The link is active
*An inactive link means that the password is wrong, it is an unsupported network device, a user account for netrangr does not exist (if one is needed), or some other type of problem has occurred. An inactive link means that *managed* will remove ALL configuration information pertaining to this network device during normal garbage collection operations.	

What happens if the password I configured is wrong?

If *managed* determines that the password specified is incorrect, it will generate an error message and log it to both its error file, *errors.managed* and to destinations configured in /usr/nr/etc/destinations. After the error message is generated, the connection state is set to Inactive and *managed* will delete ALL configuration information pertaining to this network device during normal garbage collection operations.

Sample error file message:

```
12/21/96 17:41:44, Wrong username/password for net device 10.1.2.1,
errno = 0
```

Sample error message reported in log:

```
2,1101604,1996/12/21,23:41:44,1996/12/21,17:41:44,10003,10,100,
Wrong username/password for net device 10.1.2.1
```

What happens if the network device specified is not supported?

If *managed* determines that a network device is unsupported, it will generate an error message and log it to both its error file, *errors.managed* and to destinations configured in /usr/nr/etc/destinations. After the error message is generated, the connection state is set to inactive and *managed* will delete ALL configuration information pertaining to this network device during normal garbage collection operations.

Sample error file message:

```
12/21/96 17:41:44, Unsupported net device 10.1.6.244, errno = 0
```

Sample error message reported in log:

```
2,1101601,1996/12/21,23:41:44,1996/12/21,17:41:44,10003,10,100,
Unsupported net device 10.1.6.244
```

What happens if the network device reboots?

If the network device reboots, *managed* will reestablish the Telnet connection once the device is again operational. When *managed* loses connectivity to the network device, it will generate an error message and log it to both its error file and configured destinations.

Sample error file message:

```
12/21/96 17:41:44, Connection lost to net device 10.1.6.244, errno = 0
```

Sample error message reported in log:

```
2,1101601,1996/12/21,23:41:44,1996/12/21,17:41:44,10003,10,100,
Connection lost to net device 10.1.6.244
```

Different timeout values are used for different network devices. *managed* notices that the connection is down if it has not received information from a BorderGuard within 45 seconds or from a Passport within 5 minutes. This is generally enough time for these devices to reboot and configure themselves.

Can I have *managed* tell a network device to execute a command?

You can use the NetDeviceCommand token to perform EXEC operations to pass commands directly to a network device over the Telnet connection that *managed* has established. The token requires the IP address of the particular network device to be accessed. Any additional arguments are passed directly on to the network device. The token only works if the Telnet connection is active. An example is shown below:

```
netrangr% nrexec 10003 10 100 1 10 NetDeviceCommand 10.1.2.1 net d app
border System 5> net d app
Apply Points:
point    filter name            protocol
first    FIRST_FILTER           IP
Filters active = 1
COMMAND_FINISHED
```

In this case, the command `net d app` is executed on a network device with an address of 10.1.2.1. The response that is passed back includes a copy of the command line, the results of the command, and a token specifying that the command is finished. *managed* will strip out any blank lines that might be returned by the command. This token can be used with a shell script to provide easy command line access to a network device. The example below, called bg1, will take any arguments to the shell script and pass them on to the specified network device.

```
#!/bin/sh
/usr/nr/bin/nrexec 10003 10 100 1 10 NetDeviceCommand 10.1.2.1 $*
```

Does *managed* have any built-in commands for network devices?

managed provides the basic functionality for reconfiguring a network device to shun (i.e., stop) all traffic flowing from a particular network address. This capability is used to stop traffic from known malicious hosts. The *sensord* process on the NSX uses this capability to provide real-time shunning of network connections. An operator may also use the same function to shun an address from the Director.

How does shunning work on a BorderGuard or Passport?

Shunning will ONLY work on BorderGuards and Passports if a file called *shun.fil* containing a filter called *shun* has been provisioned. The *shun* filter should be applied at the first filter point AFTER filter *netranger* has been called. This same configuration must be used whether the network device is operating as a bridge or a router. It will even work if the device is operating as BOTH a bridge and a router and the filter is applied on the IP first filter point and the bridge first filter point.

Here is an example of a filter first_filter that is applied on the first filter point:

```
filter first_filter
   filter netranger;
   filter shun;
end
```

managed will replace the shun filter by replacing shun.fil. However, an initial filter MUST be provisioned that contains a do-nothing filter called *shun.* Here is an example in the recommended format:

```
filter shun
   any continue;
end
```

What command do I send to *managed* to shun an address?

You can use the ShunHost token to perform an **exec** operation to add an address to the list of shunned addresses. *managed* will configure ALL network devices with active connections to shun this address. A value must also be provided to specify the number of minutes for the address to be shunned. Inserting a value of 1000000 (~ 2 years) effectively shuns an address permanently.

After the number of minutes has expired, *managed* will reconfigure the network devices to unshun the address. If multiple shun commands are sent for a single address, *managed* will keep updating the time value for when the address can be unshunned.

Here is an example of a successful shun command:

```
netrangr% nrexec 10003 10 100 1 10 ShunHost 10.1.1.5 1024
Success
```

managed will reject any address that contains a value of 0 or 255 as one of the four address segments. Examples of addresses that would be rejected include 10.1.1.0, 10.1.0.1, 10.1.1.255, 0.0.0.0, and 10.255.255.255.

How do I obtain a list of shunned addresses?

You can use the ShunHostList token to perform a **get** operation to return a complete list of all currently shunned addresses and the number of minutes left for each one to be shunned. If there are no addresses in the list, an error message will be returned stating that the value was unset.

```
netrangr% nrget 10003 10 100 1 ShunHostList
10.1.1.5 1024
```

How do I unshun an address?

You can use the UnshunHost token to perform an **exec** operation to remove an address from the list of shunned addresses. If the address is within the list, ALL network devices will be reconfigured to unshun the address.

```
netrangr% nrexec 10003 10 100 1 10 UnshunHost 10.1.1.5
Success
```

Can I shun a range of addresses?

Yes, the tokens ShunNet, UnshunNet, and ShunNetList provide the same functionality as the tokens previously mentioned for hosts except that they address entire networks. These tokens are designed to provide network security analysts the ability to completely shun an offending network.

The primary difference for these tokens is that a netmask must also be specified along with the IP address. Here are some examples:

```
netrangr% nrexec 10003 10 100 1 10 ShunNet 10.1.1.0 255.255.255.0 1024
Success

netrangr% nrget 10003 10 100 1 ShunNetList
10.1.1.0 255.255.255.0 1024

netrangr% nrexec 10003 10 100 1 10 UnshunNet 10.1.1.0
Success
```

Note that there are two tables of shunning information maintained by *managed*—one for individual addresses and the other for network addresses.

How do I configure *managed* to NEVER shun a particular address?

You can use the NeverShunAddress token to perform **getbulk**, **set**, and **unset** operations to manage a list of IP addresses and netmasks that *managed* compares with addresses to be shunned. If the shunned address falls within the range of addresses specified by the list, *managed* will NOT shun that address on the network devices.

Addresses may be individual hosts or entire networks as specified with the standard IP netmask. For example, 10.1.100.1 255.255.255.255 would define a single host and 10.1.1.0 255.255.255.0 would define an entire class C network.

```
netrangr% nrset 10003 10 100 1 NeverShunAddress 10.1.100.0 255.255.255.0
Success

netrangr% nrgetbulk 10003 10 100 1 NeverShunAddress
10.1.100.0 255.255.255.0
10.1.1.201 255.255.255.255
```

WheelGroup recommends that the configuration file for *managed* contains NeverShunAddress entries for the NSX, the network device, and the Director.

What is the hierarchy of shunned addresses, networks, and never shun addresses?

When *managed* receives a request to shun an IP address, it first checks to determine if the address falls within the ranges of addresses specified to never shun. It then checks to see if the address falls within the ranges of network addresses that are currently being shunned. *managed* will only apply the shun if the address is not within either set. If the address has already been shunned, *managed* will simply update its internal timer to remove the shun against that address. Shunning network addresses follows a similar pattern.

The difference between an address not being shunned because it is not supposed to be or an address that is already shunned is in the response. If an address falls within the range of addresses that are never shunned, an error message is returned. If an address falls within the range of network addresses that are already shunned, a success message is returned.

When either list of shunned addresses or networks changes, *managed* will modify the *shun* filter on the network devices to reflect the change.

Can I unshun all shunned addresses and networks at once?

Use the UnshunAll token to perform an **exec** operation to unshun ALL currently shunned addresses and networks. If NeverShunAddress entries have not been inserted for the NSX, Director, and network device—and one of them has been shunned—this command may fail.

How fast does shunning occur?

Shunning a host or network usually occurs within 3 seconds or less. It may take longer if a large number of addresses (> 100) are being shunned. This is due to the speed of the storage device on the network device. The new shun filter needs to be written to the filesystem, and then read in and compiled. The list below specifies the types of filesystems used:

```
BorderGuard 1000 - Flash memory
BorderGuard 2000 - Floppy disk
Passport - Hard drive
```

Is there a limit to the number of addresses/networks that can be shunned?

Use the ShunMaxEntries token to configure the maximum number of addresses that may be shunned. The default value is 100, but this may be increased if required. Caution is advised when increasing this limit. The limit exists for several reasons:

- The file systems of network devices do not have unlimited storage space.
- Every address shunned has a slight performance impact on the routing/bridging speed of the network device. A few addresses have a negligible impact; two hundred would have a measurable impact.
- Denial-of-service attacks may be implemented that cause NetRanger to shun large numbers of addresses. All network security solutions are vulnerable to these types of attacks. These attacks may be compared to an adversary launching hundreds of diversionary attacks to draw attention from the real threat.

The following examples illustrate how to increase the limit:

```
netrangr% nrset 10003 10 100 1 ShunMaxEntries 150
Success

netrangr% nrget 10003 10 100 1 ShunMaxEntries
150
```

What happens to the shunned configuration if the network device reboots?

When the network device reboots, *managed* will reapply the shun information when the device is operational. Systems like the BorderGuard will reestablish shunning because it will compile the *shun.fil* file modified by *managed* when it shunned an address.

Can I disable/enable shunning?

Use the ShunEnable and ShunDisable tokens to perform **exec** operations to enable or disable shunning. When *managed* is first started, shunning is always enabled by default. Shunning can be disabled by using the ShunDisable token. The following examples illustrate how shunning may be disabled and then re-enabled:

```
netrangr% nrexec 10003 10 100 1 10 ShunDisable
Success

netrangr% nrexec 10003 10 100 1 10 ShunEnable
Success
```

All shunning information will continue to be collected by *managed* during the period that shunning is disabled. Once shunning is re-enabled, then *managed* will update the *shun* filters on the network devices.

When would I want to disable shunning?

Shunning is accomplished through a Telnet session that *managed* establishes to the network device. Shunning expects that the connection be at the same prompt as at initial login. This is of primary concern for the Passport. Provisioning mode may be entered by sending the command **nrexec ... start prov**. The Telnet session will then be in provisioning mode and shunning will fail. If *managed* is used to provision a Passport, it is HIGHLY recommended that shunning be disabled during that period.

How do I use *managed* to manage the time on the system?

The following list of tokens may be used to get/set the time. In order to change the current time, *managed* MUST be running with root privileges. *managed* expresses time as the number of elapsed seconds since 00:00 GMT, January 1, 1970.

Token	Operation	Description
CurrentTime	`get`	Gets the current time
SetCurrentTime	`exec`	Sets the current time
AdjCurrentTime	`exec`	Adjusts the current time

Changing the time with a delta of greater than +/- 60 seconds may cause problems in running NetRanger processes. In this case, it is recommended to stop/start the affected processes for proper operation. Smaller changes should be accomplished through the use of the AdjCurrentTime token discussed below.

AdjCurrentTime adjusts the system's notion of the current time, as returned by CurrentTime, advancing or retarding it by the amount of time specified in the `struct timeval` pointed to by delta.

The adjustment is affected by speeding up (if that amount of time is positive) or slowing down (if that amount of time is negative) the system's clock by some small percentage, generally a fraction of one percent. Thus, the time is always a monotonically increasing function. A time correction from an earlier call to AdjCurrentTime may not be finished when it is called again.

These commands may be used in NetRanger Directors that synchronize the clocks of computers in a local area network. Such time servers would slow down the clocks of some machines and speed up the clocks of others to bring them to the average network time. This could be accomplished through the use of cron jobs and shell scripts to synchronize remote NSX systems.

How should the netrangr user account be configured on a Passport?

The netrangr user account should be configured with the following characteristics:

Userid:	NETRANGER
Scope:	network
impact:	debug
nmifs:	local, Telnet, fmip, FTP
Directory:	/

Can *managed* write out other files besides its own configuration file?

The tokens WriteFileHosts, WriteFileServices, WriteFileAuths, and WriteFileDestinations can be used to write out the configuration files in the /usr/nr/etc directory. All NetRanger daemons cache the contents of these files in memory upon startup and refresh the information when they reread their configuration file (via a HUP signal or **exec** token ReadFileConfig). ONLY *managed* has the ability to write out these files. The files would be written out if the information cached in memory was modified using the applicable configuration tokens.

What is the list of *managed* specific tokens?

Token	GET	GETBULK	SET	UNSET	EXEC
NetDevice	X	X	X	X	
NetDeviceStatus	X	X			
NetDeviceVersion	X	X			
NetDeviceType	X	X			
NetDeviceCommand					X
NumberOfNetDevices	X				
ShunMaxEntries	X		X		
ShunEnable					X
ShunDisable					X
ShunHostList	X				
ShunHost					X
ShunNetList	X				
ShunNet					X
UnshunHost					X
UnshunNet					X
UnshunAll					X
NeverShunAddress		X	X	X	
CurrentTime	X				
SetCurrentTime					X
AdjCurrentTime					X
WriteFileHosts					X
WriteFileServices					X
WriteFileAuths					X
WriteFileDestinations					X

Index

A

adding Director entities 4-10
alarms 1-9
alternate loading procedures 5-42
alternate routes 1-11
atomic signature 1-8
attack signatures 1-5, 1-8
auths E-16

B

basic Director functions 4-3
BorderGuard
  BorderGuard installation 3-5
  configuration 3-5
  hardware components F-3
  installation options
    bridge 2-3
    Internet router 2-5
    introduction 2-3
    unable to replace existing Internet router 2-6
      if no unused Class C addresses exist 2-8
      unable to subnet to existing Class C address 2-9
      use a Class B address 2-6
      use an unused Class C address 2-6
  manual reconfiguration for shunning 1-9
  physical and operational specifications 2-13—2-15
  standard files to configure B-1
BorderGuard 1000 1-7
BorderGuard 2000 1-7
BorderGuard installation 3-5—3-9
bridge mode B-7
bridging
  and BorderGuard installation 2-3
  and Data Privacy Facility B-13

C

cable requirements (BG1000/2000 to NSX) 3-7
centralized command and control 1-14
changing IP addresses, hostnames, and NetRanger IDs 4-28
changing the event error notification user 4-60
checking the status of the Director processes 4-5
commands
  exec 1-18
  get 1-17
  getbulk 1-17
  set 1-18
  unset 1-18
communication
  alternate routes 1-11
  based on three-part address system 1-11
composite signature 1-8
configd E-3
configuration commands
  nrexec E-7
  nrget E-6
  nrgetbulk E-6
  nrset E-6
  nrunset E-6
configuration of netrangr user account H-14
configuring an application with nrConfigure 4-50
configuring new users 3-25
configuring the user interface
  HP-UX and Sun Solaris 3-21
  IBM AIX 3-24
content-based signatures 1-6
context-based signatures 1-6
copy_to/log_to auditing 1-7
creating a trouble ticket 4-28

D

daemon configuration files E-8
daemons
  applications E-2
    configd E-3
    eventd E-3
    loggerd E-3
    managed E-4

postofficed E-4
sapd E-5
sensord E-5
smid E-5
as fundamental design principle 1-4
daemon configuration files E-8
managed.conf E-9
sensord.conf E-10
smid.conf E-13
daemons at startup E-17
determining the version of a remote daemon 4-26
eventd 1-20, 4-55—4-60
advanced setup
changing the event error notification user 4-60
monitoring multiple organizations 4-59
basic setup 4-56
event script setup 4-56
eventd.conf setup 4-58
overview 4-55
smid.conf setup 4-58
postofficed 1-11
remote configuration 4-23
Data Privacy Facility 1-3
and bridging B-13
filters B-10
pubkeys.dpf B-10
sleeves.cmd B-12
sleeves.dpf B-11
DBMS setup checklist D-8
defining security policy 3-27
deleting Director entities 4-19
destinations E-16
Director
adding entities 4-10
manually adding a Director symbol 4-13
manually adding an application symbol 4-10, 4-12
manually adding an NSX collection 4-14
manually adding an NSX machine symbol 4-10
basic Director functions 4-3—4-5
starting the Director 4-3
stopping the Director 4-4
centralized command and control 1-14
changing IP addresses, hostnames, and NetRanger IDs 4-28
changing registration files 4-29—4-34
changing symbol, object, and submap characteristics 4-36—4-47
changing the number of events displayed in event list 4-35

checking the status of the Director processes 4-5
command line parameter examples 4-35
configuring new users 3-25
configuring the user interface
HP-UX and Sun Solaris 3-21
IBM AIX 3-24
creating a trouble ticket 4-28
deleting entities 4-19
hardware and software requirements 3-10
installation 3-17
introduction 1-3, 4-1
limiting access to security information 4-44
maps
changing configuration parameters 4-21
creating and using multiple maps 4-41
partitioning 4-20
samples 4-46—4-47
setting a home submap 4-38
setting map groups and permissions 4-46
submap modifications 4-37
using read-only maps 4-43
modifying and viewing entity attributes 4-15—4-19
NetRanger Director user interface menu functions 4-23
determining the version of a remote daemon 4-26
remotely configuring NetRanger daemons 4-23
resolving an alarm's IP address 4-26
shunning IP addresses and Class C IP networks 4-26
unshunning IP addresses and Class C IP networks 4-27
viewing alarm context information 4-23
viewing database status and configuration 4-25
viewing event lists 4-24
network security maps 1-15
NSX management 1-16
NSX monitoring 1-15
post-installation 3-19
configuring NetRanger background processes 3-19
configuring network management background processes 3-20
pre-installation
HP-UX and Sun Solaris 3-12
IBM AIX 3-15
saving object data to a file 4-27
setting user groups 4-45
setup options 2-13
starting the Director 4-3
starting the NetRanger background processes 4-3
starting the network management background processes 4-3

starting the network management user interface 4-4
stopping the NetRanger background processes 4-5
stopping the network management user interface 4-4
submap hierarchy 4-6—4-9
submap modifications 4-37
symbol modifications 4-36
symbols
hiding 4-39
repositioning 4-39
searching for symbols 4-37
troubleshooting
Director not running A-2
Director running A-4
uninstallation G-1

E

encrypted sleeve
capabilities 1-13
introduction 1-3
event signatures 4-60—4-82
IP options events 4-65—4-67
TCP/IP event signatures 4-63
content-based signatures 4-64
context-based signatures 4-63
eventd 4-55—4-60
advanced setup
changing the event error notification user 4-60
monitoring multiple organizations 4-59
applications E-3
basic setup 4-56
event script setup 4-56
eventd.conf setup 4-58
overview 4-55
smid.conf setup 4-58
and user-defined actions 1-20
eventd.conf setup 4-58

F

File 5-46
FileMgmt tokens 5-46
filter points 1-7
installation of filters 3-8

filter templates B-3
filters
  DPF B-10
  first.fil B-3
  incom(X).fil B-3
  installing under NetSentry 3-8
  last.fil B-6
  shun.fil B-6
filters.cmd B-2
first.fil B-3
  default setting 1-7

H

hierarchies and message propagation 1-12
hosts E-15
how to partition a map 4-20

I

ICMP events 4-67—4-78
incom(X).fil B-3
install command E-7
intrusion detection engine 1-2
intrusion detection signatures C-2

L

last.fil B-6
limiting access to security information 4-44
local installation of Oracle D-2
loggerd
  applications E-3
  NumberOfSwitchBytes 5-7
  NumberOfSwitchMinutes 5-7
  SwitchFile 5-9

M

managed
  applications E-4
  configuration H-1

location H-1
managed.conf E-9
and network devices H-1—H-7
shunning H-7—H-12
tokens H-15
writing files H-14
managed.conf E-9
maps
changing configuration parameters 4-21
creating and using multiple maps 4-41
partitioning 4-20
setting a home submap 4-38
setting map groups and permissions 4-46
submap modifications 4-37
message propagation 1-12
modifying a submap 4-37
monitoring multiple organizations 4-59

N

NetRanger
architecture 4-2
bridging B-7
default alarm settings C-1—C-8
Director (see Director)
intrusion detection signatures C-2
physical installation considerations 2-13
power requirements 2-13
pre-installation 2-1
token order in nrConfigure 4-51
verify NetRanger/Oracle connectivity D-7
NetRanger Configuration Instructions and Worksheets 3-26
NetRanger installation
analyze current network architecture 2-1
BorderGuard/Passport installation (see BorderGuard)
configuration
ICMP 3-28
TCP 3-29
UDP 3-31
define all entry and exit points 2-1
defining security policy 3-27
Director installation (see Director)
and firewalls 2-3
gathering network and security information 3-28
NSX installation (see NSX)

and security filters in existing routers 2-3
NetSentry 1-7
filter installation 3-8
signatures 4-81
NetSentry-based signatures 4-81
network protocols 1-6
new signatures 4-61
non-Oracle install and setup D-7
nr_log_alarm 5-38
nr_log_alarm_1 5-38
nr_log_command 5-40
nr_log_context 5-39
nr_log_error 5-40
nr_log_tcpconn 5-39
nrConfigure 1-17, 4-48—4-54
architecture 4-48
configuring an application 4-50
help 4-50
overview 4-48
quitting 4-49
starting 4-49
token order 4-51
nrdirmap
disabling 4-44
finding out about nrdirmap 4-27
nrdirmap's command line parameters 4-30—4-35
-a, Alarm consolidation threshold 4-30
-c, Critical value threshold 4-30
-d, Disable nrdirmap by default for new maps 4-31
examples 4-35
-f, force Full synchronization 4-31
-i, seconds to be Idle before sleep 4-32
-k, Keep symbol copies during delete 4-32
-m, Marginal value threshold 4-32
-o, maxiumum number of Objects represented in a map 4-32
-p, Propagate to IP Map 4-33
-s, Secure map creation 4-33
-t, Tracing enabled 4-34
nrstart E-7
nrstatus E-7
nrstop E-7
NSX
attack responses 1-8
attack signatures 1-5
capabilities 1-5
configuration

sysconfig-nsx utility 3-4
configuration commands E-6—E-7
daemon applications E-2
configd E-3
eventd E-3
loggerd E-3
managed E-4
postofficed E-4
sapd E-5
sensord E-5
smid E-5
data collection and SAP 1-18
event notification 4-1
file structure E-1
~/bin E-2
~/classes E-8
~/etc E-8
~/tmp E-19
first.fil 1-7
hardware components F-1
information generation 1-11
installation 3-1
attach cables 3-2
default root password 3-4
initial access and system configuration
console access 3-3
modem access 3-3
network access 3-2
serial access 3-3
logging in 3-4
position the NSX 3-1
installation options
installation on corporate network 2-11
installation on isolated network 2-10
installation on switched Ethernet network 2-12
introduction 2-9
introduction 1-2
intrusion detection 1-7
logging 1-9
physical and operational specifications 2-15
and real-time intrusion detection 1-7
sensord alarm generation 1-9
system commands E-7
system files E-14
auths E-16
daemons E-17

destinations E-16
hosts E-15
organizations E-14
routes E-17
services E-15
troubleshooting A-7
NSX 1000 1-7
NSX 2000 1-7
NSX 5000 1-7
NSX monitoring 1-15
NumberOfSwitchBytes 5-7
NumberOfSwitchMinutes 5-7

O

Oracle
create user D-6
install and setup D-1
installation and mounting D-2
local D-2
remote D-2
non-Oracle install and setup D-7
planning your environment D-1
troubleshooting A-8
UNIX setup of Oracle environment variables D-3—D-4
verify generic connectivity D-4—D-5
verify NetRanger/Oracle connectivity D-7
organizations E-14

P

packet filter devices
capabilities 1-6
packet filtering device 1-2
Passport 1-6, 1-7
configuration 3-8
configuration of netrangr user account H-14
user account installation 3-9
physical installation considerations 2-13
policy violations C-6
post office subsystem 1-3
postofficed
applications E-4
and daemon communication 1-11

pre-installation 2-1
product capabilities 1-3
pubkeys.dpf B-10

R

regular expression-based signatures 4-78
remote configuration of daemons 4-23
remote installation of Oracle D-2
resolving an alarm's IP address 4-26
routes E-17

S

sapd 5-9
  applications E-5
sapx 5-11
  alternate loading procedures 5-42
  and database load setup 5-24
  and native Oracle schemas 5-37
  parameters 5-11
Security Analysis Package (SAP)
  data analysis 1-19
  fail-safe features 5-8—5-9
    file loss from overwriting 5-9
    runaway serialization 5-8
  function, configuration, and usage 5-4
  introduction 1-3
  log file serialization 1-9
  push-pull mechanism 1-18
Security Management Interface (SMI) 1-3
security policy, defining 3-27
sensord
  alarm generation 1-9
  applications E-5
  and IP Session logs 1-10
  shunning 1-9
sensord.conf E-10
serialization of log files 1-9
services E-15
shun.fil B-6
shunning 1-6
  manual reconfiguration of BorderGuard/Passport 1-9
  and sensord 1-9

shunning IP addresses and Class C IP networks 4-26
unshunning IP addresses and Class C IP networks 4-27
signatures
atomic 1-8
attack 1-5
composite 1-8
content 1-6
context 1-6
intrusion detection C-2
NetSentry-based 4-81
new 4-61
regular expression-based 4-78
TCP/IP content-based signatures 4-64
TCP/IP context-based signatures 4-63
skel 5-41
sleeves.cmd B-12
sleeves.dpf B-11
smid
applications E-5
smid.conf E-13
smid.conf setup 4-58
SQL
database information 5-37
default schemas 5-37
starting the Director 4-3
starting the NetRanger background processes 4-3
starting the network management background processes 4-3
startup file B-2
stopping the Director 4-4
stopping the NetRanger background processes 4-5
stopping the network management user interface 4-4
submap modifications 4-37
SwitchFile 5-9
sysconfig-nsx utility 3-4
system commands
install E-7
nrstart E-7
nrstatus E-7
nrstop E-7

T

TCP/IP event signatures 4-63
tokens
FileMgmt 5-46

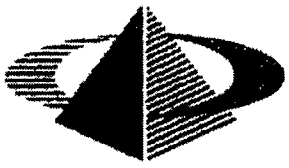
managed H-15
troubleshooting
connectivity A-6
Director not running A-2
Director running A-4
NSX A-7
Oracle A-8

U

uninstalling the Director G-1

V

verify NetRanger/Oracle connectivity D-7
viewing alarm context information 4-23



WheelGroup Corporation

WheelGroup Corporation welcomes your ideas on how to improve our documentation. Please take a moment to answer the questions below by checking either the "Yes" or "No" box. Explain any "No" responses in the sections provided. Include other comments in the "Comments" section. When you have finished, please detach and mail this page, or fax it to (210) 494-6303. You may also e-mail us about documentation issues at documentation@wheelgroup.com.

Does this manual contain all the information you expected?	◊ Yes	◊ No
Are all documented procedures in the manual correct?	◊ Yes	◊ No
Is this manual easy to read and understand?	◊ Yes	◊ No
Are the examples and diagrams helpful?	◊ Yes	◊ No
Are there enough examples and diagrams?	◊ Yes	◊ No
Other comments:		