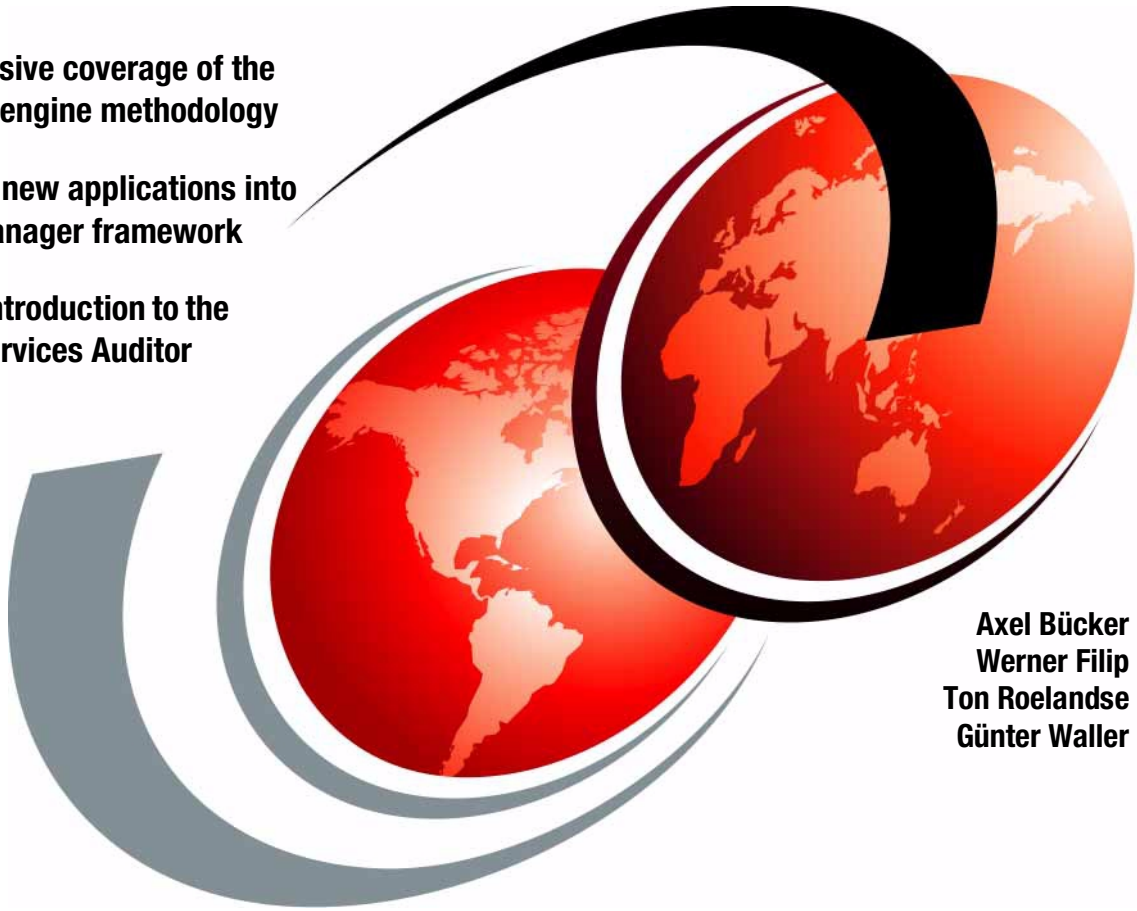
Tivoli

IBM

Tivoli SecureWay Risk Manager

Correlating Enterprise Risk Management

- Comprehensive coverage of the correlation engine methodology
- Integrating new applications into the Risk Manager framework
- Complete introduction to the Network Services Auditor



Axel Bücker
Werner Filip
Ton Roelandse
Günter Waller

ibm.com/redbooks

Redbooks

SG24-6021-00

International Technical Support Organization

Tivoli SecureWay Risk Manager
Correlating Enterprise Risk Management

November 2000

Take Note!

Before using this information and the product it supports, be sure to read the general information in Appendix C, “Special notices” on page 209.

First Edition (November 2000)

This edition applies to Version 1.0 of Tivoli SecureWay Risk Manager, for use with the Windows NT Server 4.0 or Solaris 2.7. Tivoli SecureWay Risk Manager prerequisites the Tivoli Framework Version 3.6.2 and Tivoli Enterprise Console (TEC) Version 3.6.2. The Risk Manager Adapter ISS RealSecure is running on Windows NT and the Risk Manager Adapter for Cisco SecureIDS is running on Solaris 2.7.

This document created or updated on November 13, 2000.

Comments may be addressed to:
IBM Corporation, International Technical Support Organization
Dept. OSJB Building 003 Internal Zip 2834
11400 Burnet Road
Austin, Texas 78758-3493

When you send information to IBM, you grant IBM a non-exclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

Contents

Figures

Tables

Preface

In the emerging world of e-business and worldwide internet connectivity, organizations have to protect themselves from intrusion into their online assets, as it might prove disastrous if an attacker succeeds in damaging or stealing valuable resources. Increasingly, attacks and intrusions target the enterprise as whole, not just a sub-system. Consequently, defending against these attacks requires an enterprise view of security; a coordinated approach that can harness the intelligence across different security checkpoints within the Enterprise.

There are many different potential breaches in the networking IT infrastructure. To monitor the complete spectrum, an organization needs to deploy a number of different toolsets. The number and variety of individual reports and warnings, including false alarms, generated make it difficult to monitor the entire system.

Tivoli SecureWay Risk Manager is the industry's first Enterprise Risk Management product. Risk Manager is an open, cross-platform, standards-based enterprise scale management platform that enables customers to seamlessly manage security intrusions and vulnerabilities across networks, hosts, operating systems, applications, servers, and desktops.

Tivoli SecureWay Risk Manager will cover the gamut of enterprise security systems including firewalls, routing infrastructure, network and host-based intrusion-detection systems, host-system security, anti-virus systems, and Desktop and content security. Using the Tivoli Enterprise Console, customers can centrally monitor, correlate, manage, and respond to firewall alerts, intrusion-detection alerts, virus alerts, unauthorized access, suspicious activities, policy violations, etc.

This redbook helps you understand the Risk Manager architecture and correlation process. It will also cover the integration of new detection sensors into the Risk Manager environment. An additional tool, the Network Services Auditor (NSA), will be described in detail. NSA will help security officers and administrators to investigate for specific vulnerabilities inside their network and check for results in Risk Manager.

The team that wrote this redbook

This redbook was produced by a team of specialists from around the world working at the International Technical Support Organization Austin Center.

Axel Bücker is a Senior Software I/T Specialist at the International Technical Support Organization, Austin Center. He writes extensively and teaches IBM classes worldwide in areas of Software Security Architecture and Network Computing Technologies. He holds a degree in computer science from the University of Bremen, Germany. He has 14 years of experience in a variety of areas related to Workstation and Systems Management, Network Computing, and e-business solutions. Before joining the ITSO in March 2000, Axel was working for IBM in Germany as a Senior I/T Specialist in Software Security Architecture.

Werner Filip is a Professor at the University of Applied Sciences Gelsenkirchen, at Bocholt, Germany, where he is doing research on Systems and Network Management and Applied Security. In the past he worked almost 25 years for IBM as a Consultant in Systems and Network Management at IBM's European Networking Center, Germany. He holds a degree in Mathematics and a Ph.D. in Computer Science from the Technical University Darmstadt, Germany.

Ton Roelandse is a Senior Enterprise Architect for the E.infrastructure organization within EDS, and has over 20 years of experience in the information technology industry. Currently focused on security in the e-commerce environment, Mr. Roelandse was responsible for the design and implementation of various components of a network operations center, and created and managed several help desks and support centers. Mr. Roelandse has consulted on a number of major assignments to establish network systems management practices and procedures. He implemented large help desks within major organizations in the Netherlands, the United Kingdom, Australia, Brazil, Peru, the United States, Canada, South Africa, Argentina, and Japan.

Günter Waller is a Senior I/T Specialist with Tivoli in Germany, working as a pre-sales consultant for the Tivoli SecureWay product family. He holds a degree in mathematics from Johann Wolfgang Goethe-University in Frankfurt/Main, Germany. He has 21 years of experience in various I/T areas such as mainframe systems programming, software development, and networking hard- and software. His areas of expertise include campus networking and remote LAN access. He has written and taught about the IBM 8235 Remote LAN Access Server.

Thanks to the following people for their invaluable contributions to this project:

IBM Austin
Ernest A. Keenan

Tivoli Systems Risk Manager Product and Development Team
Steve Black, Mike Garrison, William Harrison, Greg Hess, Roy Janik, Luca Loiodice

IBM Research, Zurich Research Laboratory, Switzerland
Marc Dacier, Andreas Wespi

IBM Security Analysis Team, Watson Research Center, Hawthorne
Douglas Lee Schales

IBM Security and Privacy Services, IBM Canada
David Gamey

IBM Internet Technical Support Team, Dallas
Melvin Jennings

Comments welcome

Your comments are important to us!

We want our Redbooks to be as helpful as possible. Please send us your comments about this or other Redbooks in one of the following ways:

- Fax the evaluation form found in “IBM Redbooks review” on page 223 to the fax number shown on the form.
- Use the online evaluation form found at `ibm.com/redbooks`
- Send your comments in an Internet note to `redbook@us.ibm.com`

Part 1. Introduction to Risk Management

Chapter 1. Security breaches and attacks

This chapter details where today's security risks are coming from and what sort of attacks can occur to show that intrusion detection systems must be applied to all levels within the enterprise and security point products, like firewalls, alone are not enough to ensure a secure I/T environment.

1.1 Introduction

In today's complex network environments a single change can compromise the whole network integrity and have dire financial consequences. The risk is even greater when internet transactions are involved as in the Business to Business (B2B) and Business to Consumers (B2C) models, and their derivations. A firewall alone is not enough anymore; more sophisticated mechanisms on many different levels within the company must be used to detect intrusion.

1.1.1 Reliability of a firewall

The pace of upgrades, patches, and new protocols for firewalls is staggering, and although it seems positive to have this progress, this is actually to the advantage of the skilled and experienced hacker. Before installing patches, upgrades or new protocols, they must be thoroughly tested. It can take weeks if not months to introduce a patch into the production environment. The hacker community is aware of this, and will exploit known bugs and security holes in your firewall software.

Note

One well known security hole is the fact that firewalls generally allow Domain Name Service (DNS) traffic on port 53 without logging or checking of the data. Hackers could exploit this by disguising their packets as DNS traffic to get past the firewall.

Please refer to Chapter 1.2, “Hacker profiles” on page 6 for a discussion of our use of the term ‘hacker’.

1.1.2 Internal threats

The likelihood of compromising security from within your organization is a lot higher than breaches from outside your organization. Research indicates that 60% to 80% of all security breaches have been caused by company insiders. The severity of a breach like this is often higher because people who are

familiar with the company's infrastructure are able to do a lot more damage in a short period of time. A firewall is not going to help in this case, but more sophisticated intrusion detection software in conjunction with state of the art management software and centralized management functions can help detect malicious attempts from within the organization.

1.1.3 External threats

Although internal threats are more likely, the idea of having people from outside, and, in fact, from all over the world trying to break into your company's infrastructure is a far more frightening thought. Prosecution is in many cases difficult because of the international nature of the hacker community.

1.1.4 Common forms of attack

Although the attacks and techniques are changing daily, there are a few common forms that we would like to touch on briefly:

1.1.4.1 Internal intruders

Internal intruders take advantage of sloppy administrators, non-existent or poorly designed business processes, and lack of security management.

Some common breaches are:

- Searching for shared drives in the Windows Network Neighborhood that are not password protected
- Downloading scripts from the Internet to crack passwords or identify weak passwords
- Installation of trojan horse programs (also referred to as backdoor programs) to enable future access to files and systems
- Looking for offices with Post-It notes that have passwords and login ids written on them
- Social Engineering; the use of people skills to obtain passwords and other information
- Using personal dialup accounts after departing the company left open through sloppy procedures
- Attaching a sniffer device to the local area network

1.1.4.2 External Attacks

Some common techniques are used to obtain access to infrastructures:

- Social Engineering.
- Scanning devices connected to the internet to obtain as much information on the systems as possible using a variety of techniques. Home users are especially vulnerable because of a lack of security measures, and hackers like to place keystroke monitor trojan horse programs on these home computers to obtain passwords, VPN secret keys, and other relevant information.

> **Note**
>
> You may use your desktop computer at home to browse the internet and login to the corporate VPN to check e-mail and other business related activities. If a malicious hacker placed a trojan horse program on your desktop, he would be able to obtain the information he needs to access your corporate network.

- Spoofing of a DNS connection to redirect the information to a server controlled by the hacker.
- Using operating system information to determine version and release in order to use the known bugs and security holes. This is done by looking at the header information when doing an FTP or login attempt to a targeted server.
- Script kids are using publicly available hack scripts to change Web pages and crack passwords.

> **Note**
>
> Malicious hackers normally do not advertise that they have intruded your network.

1.1.4.3 Access points

Access points are obviously a target for external hackers, and in most cases access points are managed on a case by case basis. Security policies are also often weak or non existent. The phrases “We are secure; we have a firewall” or even “We are very secure; we have multiple firewalls” are still common, but firewalls alone do not prevent hackers from entering your enterprise I/T infrastructure.

Note

A false sense of security is more dangerous than no security at all.

Access points are either created on purpose like Dial Up connections, or without being aware of them like a development server with Internet connectivity. Networks that have been linked due to consolidation or merger could be left with no one in control of the entire I/T infrastructure for a transitional period, making them more vulnerable.

1.2 Hacker profiles

The hacker community draws a line between a 'hacker', who breaks into systems for the thrills and sense of accomplishment, and a 'cracker' who does it for malicious purposes. The term 'hacker' in this redbook is used to describe a person or groups of people who are trying to gain unauthorized access to computer systems and infrastructures. They then use the information found on the targeted infrastructure for criminal or improper use.

There are three graduations of hackers, and they all need to be treated with care and diligence:

Script kids - Youngsters who use existing scripts to gain access through the firewall, break passwords, and modify Web pages. They do not realize the legal implications, and their motivation is based more on curiosity than to create damage. Not only do they run scripts without understanding what they are doing, but they also jeopardize their own security because malicious code could be hidden in the script file. Malicious hackers can thereby gain access to the script kid's system and use it to launch attacks on large corporations!

An interesting statistic shows that 90% of all hackers are script kids.

Script kids are not considered the most dangerous hackers, but are responsible for a large number of attacks and traffic that could cause serious problems like Denial of Service (DOS) attacks and the creation of enormous numbers of events.

The script kids' ability in general does not exceed the "Scan & Attack" step with exceptions that reach the "Island" step (see Figure 2 on page 8 for more details).

Experienced Programmers - By definition, these people have broad experience in debugging code, finding bugs, and creating workarounds for

problems. They are typically skilled in a variety of programming languages, mainly C and C++. They use bug lists of applications to determine known vulnerabilities and their own creativity to obtain ways of accessing protected information by exploiting these bugs.

The majority of scripts that are available on the internet hack sites are created, modified, and enhanced by this group.

Less than 9% of all hackers fit in this category. Although the skill level is high it is not their main goal to abuse falsely gained information maliciously, but to boast their reputation and get a sense of accomplishment.

These experienced programmers usually get to the “Continue Dig” step, see Figure 2 on page 8 for more details, but usually are not able to cover their tracks completely, which is caused in part by their lack of expertise in Protocol and Intrusion Detection technology.

Protocol experts - This group of highly skilled programmers is able to break into a secure I/T infrastructure, change code, obtain sensitive information like credit card data, and leave the wire without being traced. Their excellent understanding of protocols like TCP/IP, UDP, IPsec., L2TP, VPN, encryption methodologies, Intrusion Detection techniques, firewalls, packet wrappers, and a multitude of low level programming languages make them a dangerous opponent if they decide to use their capabilities for illegal purposes.

This group is also responsible for some of the best hacker tools and concepts, but they are smart enough not to post these programs or brag about them.

Less than 1% of all people classified as hackers belong to this group.

This group is able to go all the way to the “Takeover” step, see Figure 2, and is therefore potentially the most dangerous.

The actual severity of these hacks is relative to the threat they represent, see Figure 1. Note, however, that there have been cases where script kids have caused serious damage through the use of tools created and provided by experts.

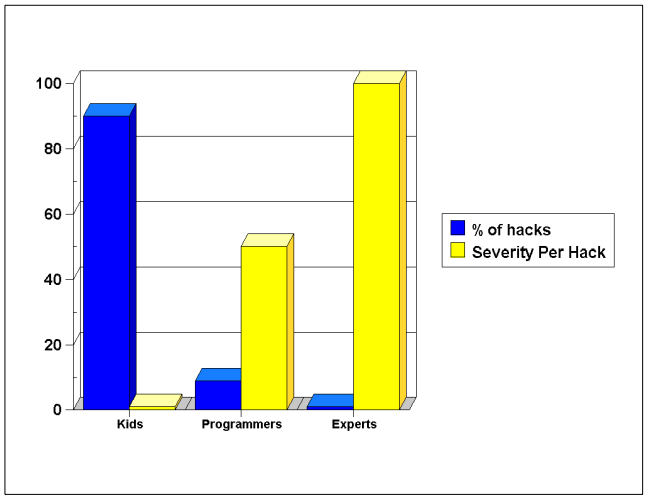
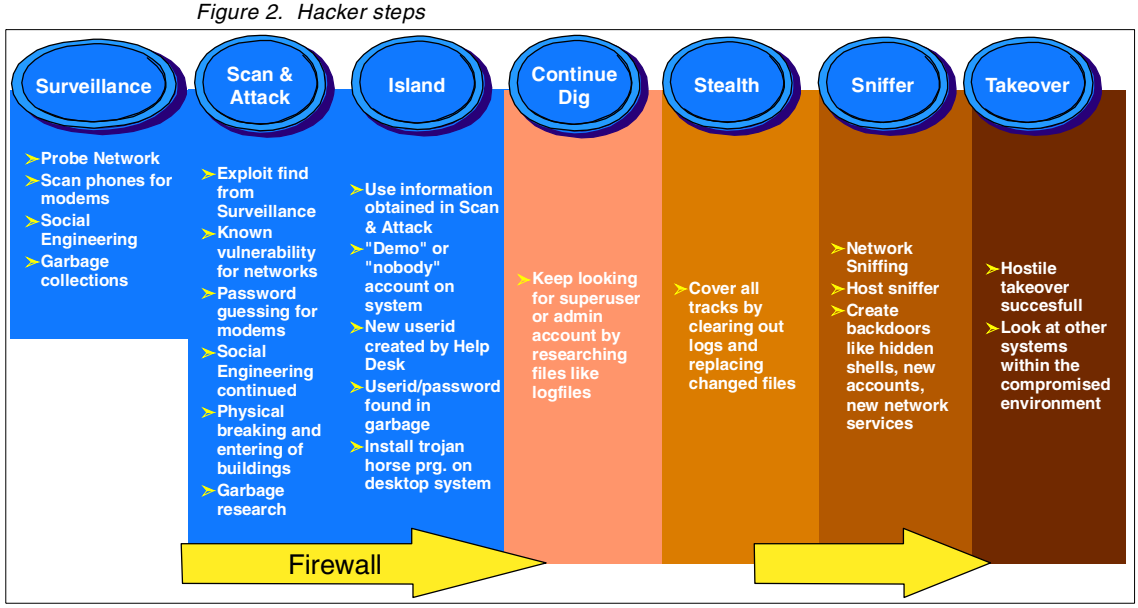


Figure 1. Percentage of Hacks relative to Severity

1.2.1 Hacker methods

The following chart is a high level overview of the steps taken by hackers with the goal to take over your network.

Figure 2. Hacker steps



This introduction shows that in order to successfully detect intrusions, we have to be able to monitor all the different components within the corporate infrastructure, and not just equip the firewall with intrusion detection software. After the firewall is breached, an intruder is able to try to gain access to even more resources.

Note

To centrally and effectively manage the different sorts of breaches and attacks in order to reconcile the necessary countermeasures, a holistic I/T security infrastructure is the only acceptable solution.

Having said that, a holistic security infrastructure approach should follow the characteristics outlined in the next section.

1.2.1.1 Holistic security infrastructure characteristics

Sure, a firewall is a good initial protection, and it stops most hackers. But once the intruder passes the firewall, he is virtually free to do as he pleases. A good intrusion detection approach has the following characteristics, listed in random order:

- All network devices report to a central repository about possible breaches. This includes but is not limited to firewalls, Web servers, DNS servers, application servers, remote workstations (home office users), and so on.
- The central repository is managed 24 hours a day and suspicious events are reacted on immediately.
- Enterprise Risk Management processes and procedures are defined, constantly updated, and adhered to by all involved employees.
- Event flooding* and false positives** can deter support staff away from actual breaches, therefore a correlation engine (see Chapter 2, “The importance of correlation” on page 11) is an absolute must.
- Implement pervasive intrusion detection by installing a sensor or agent on every device that constantly monitors for suspicious signatures or patterns, just like antivirus software vendors have been doing. This allows for constant updates to the signature database, and provides a flexible and easy way to manage an intrusion detection system.
- A holistic security infrastructure approach is necessary to make intrusion detection and management of breaches an integral part of the network topology and business practice.
- A consistent enterprise security policy is in place. All components are set up and configured in compliance with this policy.

- Conduct regular vulnerability assessment by utilizing scanners to detect and report on weaknesses, like weak or blank passwords, unused open TCP ports, and so on. This includes testing for compliance with the security policy.
- Implement centralized reporting and analysis of breaches. This should be handled by highly skilled security specialists who are physically separated from the operation and traditional network management areas.

*Events are stored in a repository, but the amount of events causes the information to become cluttered and unusable.

**An event indicating that a breach occurred but is a false alarm instead.

Chapter 2. The importance of correlation

It is not enough to focus on individual events and their occurrence. In case of a security breach, different sorts of attacks can and will occur over a period of time. Only a centralized correlation mechanism will be capable of identifying a real security threat.

Customers today have to deploy several products as part of their e-business security strategy. Products such as Web servers, firewalls, intrusion detection, and access control systems are all required to implement specialized security functions. Very often these point products do not interoperate and have to be managed and administered individually. Most point products generate copious events and false alarms that could completely overwhelm even a skilled administrator.

Because point products have a one-sided view of security, without an automated correlation engine administrators have to sift through the log file output from each intrusion detection agent and attempt a manual correlation, a process that is laborious, time-consuming, often error prone, and, last but not least, done after the event.

Correlation done as the event occurs can help limit the damage when Denial of Service (DoS) attacks are imminent by closing down the attacked device or devices in a controlled fashion.

Without centralized management and correlation, it is almost impossible to determine attack patterns, make security assessments with any degree of assurance, or respond with real-time countermeasures. An integrated security solution is most effective when firewalls, intrusion detection agents, network security, and application security solutions can work together in a coordinated fashion to minimize threats. The central management and correlation engine implements these functions:

- Receives the alert information from one or more sources
- Centrally processes the alert information based on the alert characteristics
- Centrally correlates the alert information based on correlation rules
- Centrally stores the alert information to a relational database

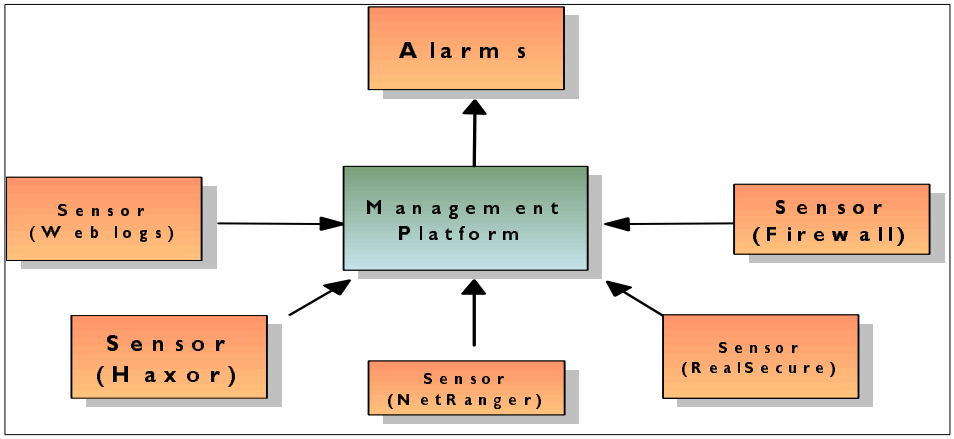


Figure 3. Grouping information sources

Figure 3 shows the grouping of information sources in a common platform. All the sensors send events to a central console that correlates similar events and presents the administrator with a consolidated output.

For the intrusion detection community, integration is more than representing the results of distributed devices on a single console. The integration of the intrusion detection systems allows correlation. A definition of correlation of intrusion detection events is given by Edward Amoroso:[1] "Intrusion correlation refers to the interpretation, combination, and analysis of information from all available sources about target system activity for the purposes of intrusion detection and response." We are going to consider the results of different intrusion detection systems as information sources. That means we do not consider all sources like network packets or log files. This is the job of the intrusion detection systems. Based on the following observations, correlation is fundamental for concise information processing:

- Commercial intrusion detection systems often generate a huge amount of data. Analyzing them directly without filtering functions is difficult and very time consuming.
- Sensors generate a high number of false alerts. For example, normal processing can result in suspicious patterns that look like attacks, which in turn increases the risk that 'real' attacks go unnoticed due to the high noise level.

[1] Edward Amoroso. Intrusion Detection. IntrusionNet Books, 1999.

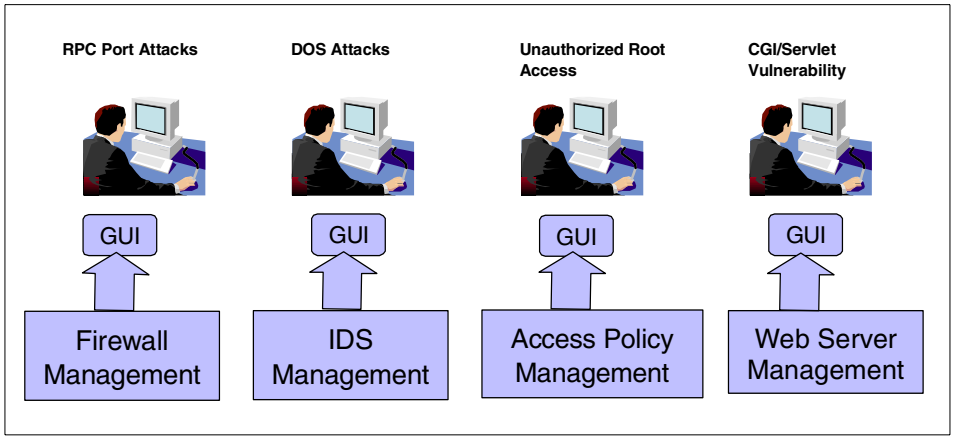


Figure 4. No integration/correlation - No control

Figure 4 shows that without integration/correlation, there is no control:

- Operator productivity is poor because of multiple consoles
- Staff cannot zero in on critical, relevant problems
- Vulnerabilities in point products are masked; hackers can attack the weakest link, making the defense useless

There is no universal intrusion detection system. Intrusion-detection systems have different strengths and weaknesses, and their information sources and processes can also be different. We can improve the detection process by combining different intrusion detection products.

Because intrusion detection systems do not have the same capabilities, they give different output. It is possible then to reduce the number of false alarms by building rules that correlate information from many different intrusion detection systems. The information can be of different types. For example, a confidence level can be associated with each alarm. The lower the confidence level, the higher the probability that the alert is false. It is also possible to increase the global attack signatures by summing up all single intrusion-detection system signatures.

Almost all events generated by host based intrusion detection systems do not concern direct attacks. They describe user actions that happen on the host. The majority of these actions should not be considered as alerts. We should take into consideration the context in which an event occurred. The common approaches analyze alarms in isolation, whereas the alarm context could

enhance decision making. The following are some examples where alarms should be triggered:

•A “failed password” happens often in every environment, and the majority of them should be ignored. However, if such an event happens in the middle of the night when the computer room is closed, an alert should be generated.

•If a strange behavior (e.g port scan) is detected on a single host, it might be a false anomaly and the security manager can ignore it. However, if such behavior is detected on several hosts in the network, someone is certainly probing the network.

•In some environments a “failed login” can occur many times a day, where as in other environments it may not be normal and should trigger an alert.

•If a system is under attack, for example an intruder has gained access, all the actions performed on the system should be reported to keep track of the intruder(s).

All these examples show that host-based events are not simple to classify. The same events can be acceptable for one company but not for another. Having an alert each time an action is performed on the host is not useful. The amount of data needs to be reduced in order to be efficient. This can be done by building rules describing the normal behavior of the system for that company.

The host-based events need special treatment to be considered as a real alarm. This can be done with a profile. The profile must include all information that can lead to a better understanding of the event’s context. The network-based intrusion-detection systems provide information about current external attacks. With this additional information you can determine if the attacker was successful or not by checking to see if the host-based intrusion detection system reports anything unusual. By monitoring critical files and correlating the information with the network-based intrusion detection system, you can tell more precisely what the attacker did and if they were successful.

Summary

Customers have made significant investments in many point products such as firewalls, intrusion detection systems and application-level security mainly because each point product implements a specific security function that is required to implement the overall security strategy. However, security is more than a firewall or other product solution. Security is a link with each point product implementing a component of the link. Each component of the security link such as firewalls, intrusion detection systems and application

security mechanisms reinforce or complement each other. The overall security is only as good as the weakest link. An integrated security management product enables customers to make the most informed security decisions by leveraging the intelligence of the various security links. Centrally correlating intrusions and vulnerabilities across these different components, provides the overall assurance that individual security components reinforce and complement each other and implement the overall business goal of managing risk against information assets.

Part 2. Tivoli SecureWay Risk Manager

Chapter 3. Risk Manager architecture

This section explains the information flow and high level architecture of Tivoli SecureWay Risk Manager and the associated products, and the components involved. The Risk Manager product version used for this book is release 1 running on Tivoli Framework 3.6.2 and Tivoli Enterprise Console (TEC) 3.6.2.

3.1 Risk Manager event flow

Tivoli SecureWay Risk Manager provides a security management system for intrusion detection sensors. In the context of this document, the term "intrusion detection system (IDS)" stands for any type of sensor that reports information related to security. In other words, Risk Manager is intended to handle any type of security related information and not be restricted to conventional concepts for an intrusion detection system. One or more sensors monitor the network and generate information in the form of events (also called alerts). These events represent suspicious activity or security related problems detected by the sensor IDS. The IDS events are picked up by Risk Manager adapter components. A Risk Manager adapter component transforms the event from its native format into an instance of a Risk Manager TEC Event Class. The class definitions and class hierarchy structure are a key part of Risk Manager. The TEC event is then sent to the TEC Event Server (see Section 3.2.1, “Component overview” on page 20 for more details).

At the TEC Event Server, the Risk Manager Correlation Component processes the incoming events, generates new TEC events, and displays appropriate events at the TEC Console. The processing performed by the Correlation Component is a second key part of Risk Manager. During processing, information from the event is aggregated and the resulting data set is analyzed for patterns.

Note

The Risk Manager Correlation Component correlates IDS events with other TEC Events to determine suspicious activity.

Suspicious activity or problems detected as a result of the search for patterns are referred to as "situations". The purpose of the Correlation Component is to evaluate intrusion detection information from multiple sensors and present relevant information in a concise format.

3.2 High level architecture

Risk Manager is an integral part of the Tivoli Framework and therefore follows all rules and concepts defined by this framework. Major advantages of the Risk Management solution are the scalability and extensibility. Therefore it takes a relatively short time period to learn the product and concepts, and existing Tivoli resources can be used to integrate this product into the existing Tivoli Framework. The Risk Manager architecture overview is a high level overview of the components involved to install and operate Risk Manager.

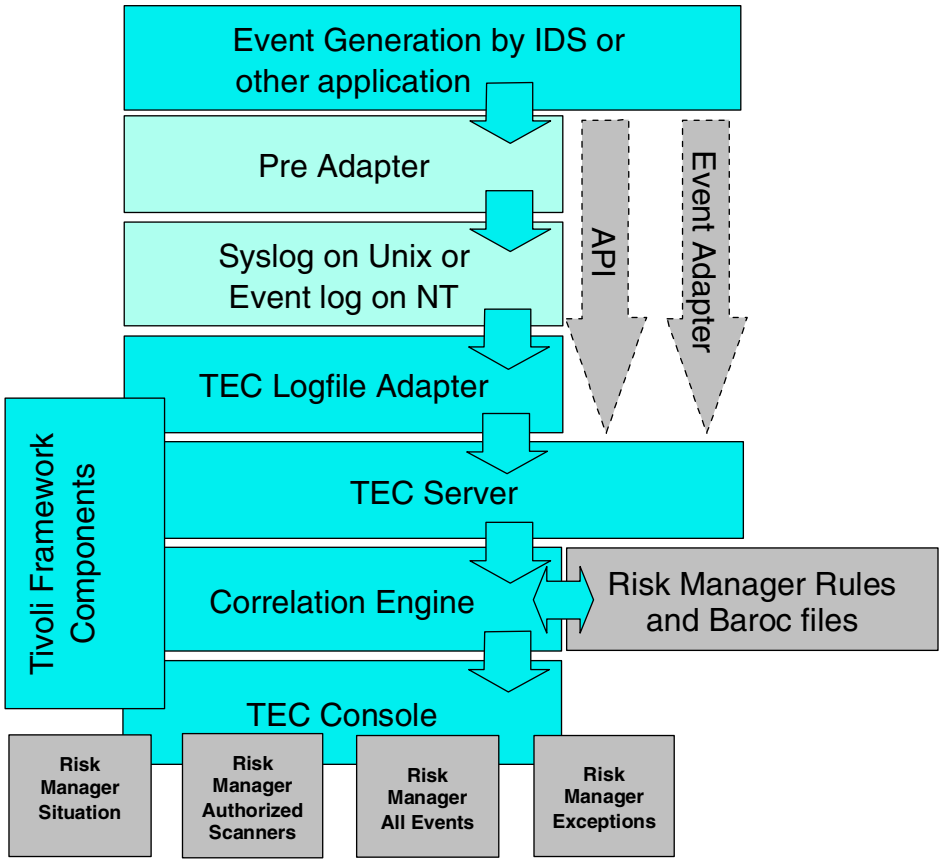


Figure 5. High Level Risk Manager Architecture Overview

3.2.1 Component overview

The Risk Manager product, because of its framework emphasis, is made up of Tivoli Framework components.

The components, as shown in Figure 5 on page 20, are:

3.2.1.1 Event generation

This layer defines the source of events sent to the TEC Engine for processing by the Risk Manager correlation rules. The Event generation layer presents third party intrusion detection system (IDS) vendors like:

- ISS RealSecure
- Cisco Secure IDS (formerly known as NetRanger)

Another type of interface is the WebIDS, which handles event signatures from Web servers such as iPlanet, Internet Information Server, and Apache.

Note

It is Tivoli's stated intention that future releases of the product will include additional event adapters for firewalls, antivirus software, operating system specific events, and more.

This layer is also the entry point for customized integration of virtually any application that is able to produce output to the Windows NT event log, the UNIX syslog, or a flat file. This is discussed in more detail in Chapter 6, "Integrating third party products into Risk Manager" on page 129.

3.2.1.2 Pre-adapter

Pre-adapters are used to retrieve and format event information from products like Cisco Secure IDS or ISS RealSecure, and store this information in either a syslog format or NT event log format. These formats enable the TEC Logfile Adapter to send these events to the TEC Server for processing.

Note

The pre-adapter component can also be substituted with a third party component to generate output to a logfile or event log. An example of this can be found in Figure 62 on page 151.

The following pre-adapters are available with the current release of Tivoli SecureWay Risk Manager:

ISS RealSecure pre-adapter

The adapter for ISS RealSecure is written in Java. Risk Manager requires and installs the IBM Java Runtime Environment (JRE) Version 1.1.8 on Windows NT with the adapter for ISS RealSecure.

The adapter for ISS RealSecure supplied with Risk Manager resides on the host where the RealSecure Management Console is located. It uses Java ODBC to interface with the RealSecure Event Database. The ISS RealSecure adapter maps these events into Windows NT event log records, which are then sent to TEC using the TEC Logfile Adapter for Windows NT. Use of the RealSecure format file enables configuration of the TME logfile adapter for Windows NT. The RealSecure adapter supports remote based UNIX logfiles to circumvent the NT event log.

CISCO Secure IDS pre-adapter
Risk Manager includes an adapter for Cisco Secure IDS that maps alarms generated by Cisco Secure Intrusion Detection System (formerly known as NetRanger) into TEC events.

The adapter for Cisco Secure IDS supplied with Risk Manager resides at the Cisco Secure IDS (NetRanger) Director host.

Web Intrusion Detection (WebIDS)
Web Intrusion Detection (WebIDS) is a Perl-based tool provided with Risk Manager. This tool uses the log files generated by a Web server to perform the analysis that detects Web server attacks. You must deploy Web Intrusion Detection at each Web server that you want to monitor.

Web Intrusion Detection uses a knowledge-based approach to detect malicious behavior. By defining general signatures of Web server attacks, a wide range of attacks can be detected. Attack signatures can be expressed as simple text strings or as Perl regular expressions. Risk Manager supplies a file that contains signatures for Web server attacks—the sig.nefarious file.

Klaxon
Risk Manager includes an adapter for Klaxon that maps alarms generated by the Klaxon IDS into TEC events. The adapter for Klaxon supplied with Risk Manager resides at the Klaxon host. The Klaxon adapter does not ship with release one of the Risk Manager product.

Note

Tivoli SecureWay Risk Manager's flexible and extensible approach allows for custom developed adapters to be designed to integrate virtually any IDS or other type of application into the Risk Manager Correlation Component. There are additional methods to get the events to Risk Manager, including wpostemsg, postemsg (non-TME), the IEF API's, and the API as discussed in Section 3.2.1.4, "API" on page 23.

3.2.1.3 Event adapter

It is Tivoli's stated intention to include the event adapter in a following Risk Manager release. It will offer another integration path besides the pre-adapter to syslog, and from the syslog to the Logfile adapter, and from the Logfile adapter to TEC. The event adapter is a single module that transports events generated by supported IDS applications directly into TEC.

The combination of Pre-adapter, Syslog, and TEC Logfile adapter can also be called an event adapter; it is composed of different components, but its purpose is the same.

3.2.1.4 API

The Risk Manager API or Common Adapter Toolkit (CAT) is designed to facilitate the following requirements for the next release of Tivoli SecureWay Risk Manager:

Some obvious advantages include but are not limited to:

- Provide common functionality to all Risk Manager adapters for sending intrusion detection events to TEC
- Provides an alternative for the use of the Tivoli logfile/NT eventlog adapters for sending Risk Manager events to TEC while still maintaining the pattern matching capabilities that these adapters support
- Allow for a standardized facility to easily integrate Distributed Monitoring Support
- Provide a Perl module interface for the existing WebIDS product and other applications
- Allow for common TME profile manager support for adapter configuration and distribution

Design strategy

The existing TME Event Integration Facility (EIF) will be used to create a simplified API set for Risk Manager Adapters to use. The Risk Manager common adapter library will also incorporate syslog / NT event Log parsing code, so that existing Risk Manager Release 1 pre-adapters will be compatible with the new common approach. This simplified approach for adapter development will allow adapters to share some commonality and decrease the development/test effort involved.

Distributed Monitoring Support

To better facilitate adding Distributed Monitoring support to the Risk Manager product, a common way of being able to query status of a Risk Manager

adapter will be designed as part of the library's API implementation. In turn, a monitoring collection will also be defined.

3.2.1.5 Syslog on Unix or event log on NT

Although they were not really components of Risk Manager, the use of these log files is still the preferred way to integrate applications into TEC because of the straightforward approach, ease of installation, and that no programming skills are necessary to perform these tasks, compared to writing an event adapter using the Risk Manager API.

More details on the NT event log and the syslog approach are described in Chapter 6, “Integrating third party products into Risk Manager” on page 129.

3.2.1.6 TEC Logfile Adapter

This adapter is available for NT and UNIX platforms, and its primary function is to collect events out of the syslogs or event logs at certain intervals and send them to the TEC server for processing utilizing a format file for data mapping. For more details please see Chapter 6, “Integrating third party products into Risk Manager” on page 129.

3.2.1.7 TEC Server

The TEC Event Server acquires information from different sources, aggregates them to automatically group raw events, generates from these raw events a very small subset of alarms, and escalates these alarms according to their severity. These events are interpreted according to the Basic Recorder of Objects in C (BAROC) definitions.

3.2.1.8 Correlation Engine

This component is the basic TEC Correlation Engine. No modifications are required when installing Risk Manager. Upon installation of the Risk Manager application, the Risk Manager Rules and BAROC files will be loaded into the correlation engine, which then forms the so called Risk Manager Correlation Component.

The TME 10 Enterprise Console rule language is an abstraction on top of the Prolog programming language. Prolog is an event-driven language that allows for the execution of combinations of operations in response to specific received events. Rules are relatively simple to create and modify.

For more information on correlation, see Chapter 4, “Risk Manager correlation” on page 27.

Note

Risk Manager correlation is a collection of Rules and BAROC files that are loaded onto the TEC correlation engine.

3.2.1.9 Risk Manager rules and BAROC files

This set of rules and BAROC files is, together with the pre-adapters, the WebIDS and the future API, the heart of the Tivoli SecureWay Risk Manager. It contains all the correlation logic needed to effectively manage your enterprise against intrusion. In the case of WebIDS, it also contains internet or intranet specific attack signatures.

3.2.1.10 TEC Console

The Standard Tivoli Framework TEC Console.

IDS events are displayed on the console, which provides multiple views of IDS activity. The event console presents a composite view of all IDS events received from IDS sensors in the network. Risk Manager creates the event groups, event filters, and the event windows (referred to as Risk Manager containers) on the event console.

3.2.1.11 Risk Manager Event Container

The Risk Manager event console uses the following containers:

- Risk Manager Situations

  Contains correlated events that after correlation are considered real threats or attacks. Events in this container need to be investigated.

- Risk Manager Authorized Scanners

  Contains known intrusion detection scanners to suppress false positive attacks caused by vulnerability tests of network scanners like NSA. For more information on NSA, see Chapter 5, “Network Services Auditor (NSA)” on page 65

- Risk Manager All Events

  A collection of all events handled by the Risk Manager correlation engine.

- Risk Manager Exceptions

  A container used to collect Risk Manager internal error events like Prolog failure, bad input, and so on.

These containers are created by the Risk Manager installation procedure as described in the *Risk Manager Installation and User’s Guide*.

Chapter 4. Risk Manager correlation

This chapter talks about the event correlation that takes place in the Tivoli SecureWay Risk Manager environment.

4.1 Introduction

Correlation makes sure that events that are critical for the security of the system appear with a high severity level and contain as much relevant information as possible in a concise format. Correlation also helps reduce the false alarm rate by making sure that there is enough significant information from different sources to certify the conclusion.

The Risk Manager security event correlation (*TEC Correlatio*n) has unique correlation rules for the Risk Manager adapter for ISS RealSecure, the adapter for Cisco Secure IDS, and the adapter for Web Intrusion Detection. However, basic correlation can be performed for additional adapters without modifying the correlation rules as long as the new adapter generates events derived from the idwg.baroc file, with all the relevant information supplied (see Chapter 4.2.4, “BAROC files” on page 35).

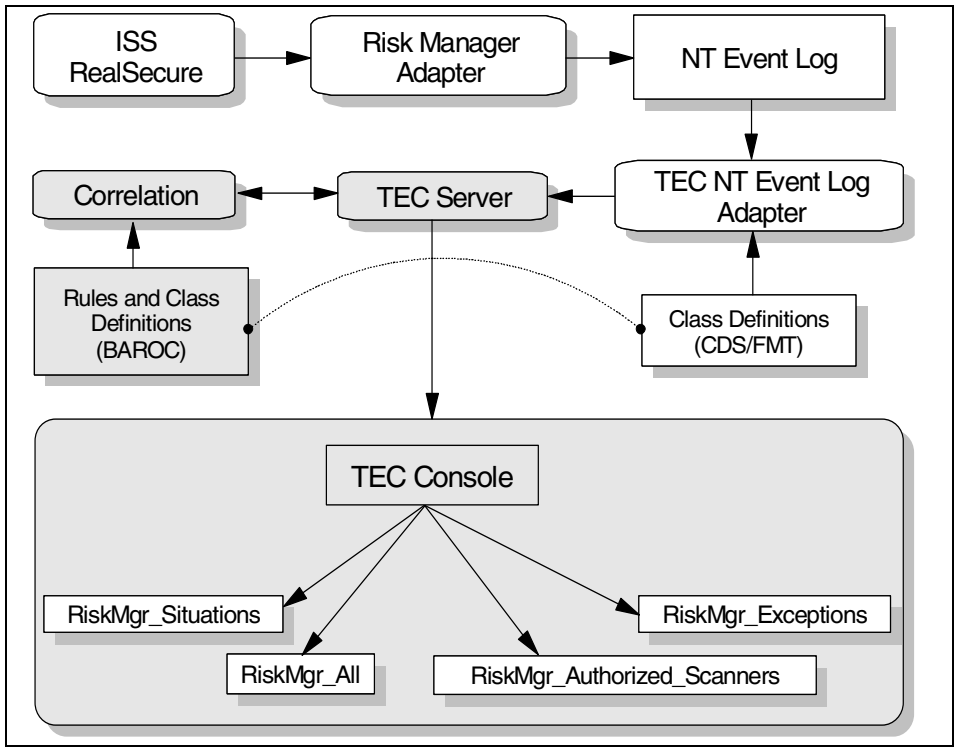


Figure 6. Data flow example: RealSecure and TEC

Before any intrusion detection system tool can be used with Risk Manager, all intrusion detection events generated by the sensors must be transformed into subclasses of the generic idwg.baroc classes that bear all the relevant information for correlation. Then the correlation rules are applied to these generic classes. The integration of any additional IDS tool requires the development of the adapter layer.

Risk Manager TEC Correlation contains rules relative to the processing and correlation of IDS events. Risk Manager uses these correlation rules to automatically group events and present a simpler picture of the security status of the network. The TEC Correlation rules are implemented using `.rls` files, written in Prolog (see Chapter 4.2.5, “Rules files” on page 37).

Risk Manager provides existing rules that are immediately available to correlate a new set of events from the correlation engine. This engine:

- Reduces the redundancy in the information presented to the operator. It is a good practice to have intrusion detection in place at several strategic locations in a company using different technologies. This involves

changing and presenting all the intrusion detection events from all the different intrusion detection systems to the user in a similar manner. Some fast intrusion detection engines send preliminary information about an attack quickly, while more sophisticated engines give more detailed information later. When it is the same attack, the information appears only one time in Risk Manager.

- Groups the data presented to the operator to avoid information overload. Attacks happen in waves, and the reaction of the operator must be prioritized. The correlation engine groups attacks hierarchically by signature, origin, and destination. The correlation engine manages the significance of the groups as well as the significance of the individual events. For example, if a probe is running from a given source, all events related to this source are grouped together.
- Manages time and confidence information. Time is an important factor in an attack and can play both ways. On one hand, old events are not relevant to most streams of attacks, and must be aged out by the reasoning process. On the other hand, long-term patterns can indicate other kinds of attacks. Time, therefore, is incorporated into the correlation engine. Also, the operator can provide feedback on the significance of events to enable you to configure the reasoning mechanism independently from the intrusion detection sensor. The sensor continues to collect data, but the TEC analyzes it and assigns a lesser or a higher value than the one originally given by the sensor. This way, you can discard known probes (such as an Emergency Response System scan) or adjust differences between tools (such as the difference of trust level between e-mail alarms given by ISS RealSecure and Cisco Secure IDS sensors).

The correlation engine contains these mechanisms:

- *Acquisition by an adapter.* Risk Manager relies on TME adapters to generate TEC events by using the appropriate format files provided. Note that any IDS sensor can forward TEC events to the correlation engine as long as it produces TEC events sub classed from the base IDS event classes.
- *Duplication check.* This phase checks for previously received events that might represent the same attack. It might be the same attack either because the intrusion detection product is sending a duplicate event or an updated event, or because another intrusion detection product has picked up the same attack and sends an event that has already been sent by the first tool.
- *Correlation with reasoning events.* After the external event has been filtered for duplicates, it is aggregated to the appropriate internal events.

- *Evaluation of reasoning events.* After the reasoning events have been updated, they are evaluated again to determine if the security status of the system has changed in a way that requires creating or updating an alarm.
- *Escalation of alarms.* When an alarm is created or updated, the alarm status is evaluated again to either:
  - Bring it more prominently to the attention of the operator
  - Degrade its severity

See Section 4.2.3, “Events and event groups” on page 33 for additional information.

4.2 Risk Manager event and data structure

The following sections describe the various event and data structures and how they are used by Risk Manger, especially by TEC Correlation during event processing.

4.2.1 Event class hierarchy

The Risk Manager event class hierarchy represents events that are generated by intrusion detection systems (IDS). This hierarchy make alerts more consistent and uniform so that correlation can be as general as possible. It acts as an abstraction layer to intrusion detection systems. The Risk Manager adapters extract the relevant information and transform it to always have the same content, regardless of the IDS source. The correlation rules are mostly applied to this abstraction layer, and are independent of each intrusion detection system.

The data structure is implemented using BAROC files that are understood by the Tivoli Enterprise Console (TEC). The idwg.baroc file implements the *generic* data structures for alerts. The rules operate on these data structures. No instances of these classes should be sent to the TEC. The other BAROC files contain the class derivatives for the alert format. The riskmgr.baroc file implements the class events.

The following diagram shows the Risk Manager event class hierarchy:

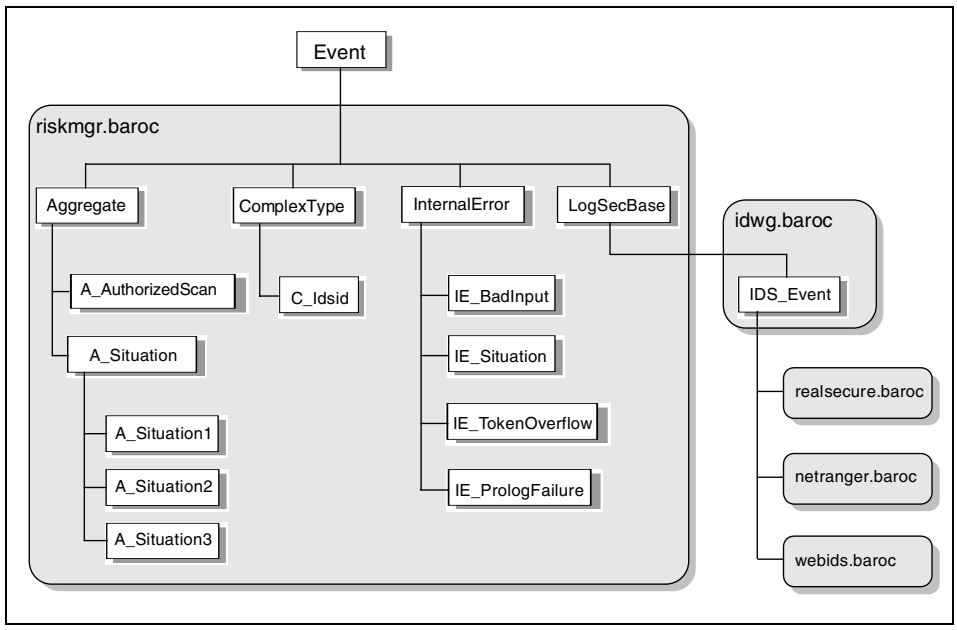


Figure 7. Risk Manager event classes

Risk Manager uses the following classes to create filters for intrusion detection event groups:

Aggregate
Use this event class as a filter to capture all alarms generated by the rules engine. It captures all situations (for authorized and unauthorized scanners).

A_AuthorizedScan
This event class relates to alarms generated from correlated idwg.baroc events that are received from authorized scanners. Authorized scanners are defined in the riskmgr_config.pro file using `authorized_scan` definitions.

A_Situation
This event class relates to alarms generated as the result of correlated events that are received from non-authorized scanners. Events can be captured with a filter using these classes: *A_Situation1*, *A_Situation2*, and *A_Situation3*.

IDS_Event
Use this class for IDS events that are received by the Risk Manager correlation engine. Events posted to this group of events are also stored in the event database.

InternalError
Use this event class for IDS events that are generated by the correlation engine when any internal errors are encountered (such as IE_BadInput, IE_TokenOverflow, IE_PrologFailure, and IE_Situation).

C_Idsid
Use this event class for IDS events that are created whenever an alien (unknown) IDS has generated an event and the following is true:

- An IDS setting is not defined in the riskmgr_config.pro for the host on which the IDS resides.
- An `ignore_ids_creation` parameter is not defined for the IDS class (such as: webids, realsecure, or netranger).

See Appendix A, “Risk Manager class description and attribute list” on page 167 for additional information.

4.2.2 Alarms

When an intrusion detection sensor detects suspicious activity, it collects the data and forwards the data to the event console. Risk Manager either logs the alarm event or generates an *alarm*. Alarms are suspicious activities that trigger administrator alarms and other configurable responses that lessen the workload of security administration. The Tivoli administrator uses the riskmgr_config.pro configuration file to define host information for rules processing and to set thresholds related to the raising of alarms.

The Risk Manager adapters map alarms generated by the commercial ISS RealSecure, Cisco Secure IDS products, or Web Intrusion Detection into *TEC event*s. *Alarm events* are created from Risk Manager reasoning events, and they are sent to the event console for operator attention.

Alarms represent the results of the correlation activity for an external event. Also, alarms are raised when TEC Correlation encounters an event from an IDS host for the first time that is not defined in the riskmgr_config.pro rules configuration file. Correlation also helps reduce the false alarm rate by making sure that there is enough significant information from different sources to certify the conclusion.

Examples of alarms that can be reported to the Tivoli user include Web scans (list of attempts), port scans (list of services), and user login attempts (list of user names).

A single alarm might represent a large number of duplicate events. It might also represent an aggregation of events, based on a variety of situations that are detected by the correlation rules.

4.2.3 Events and event groups

The TEC correlation engine recognizes four types of intrusion detection system events:

- *External events* that are created by intrusion detection tools. External events are received by the Tivoli event server for processing by the correlation engine using the Risk Manager correlation rules.
- *Reasoning events* that are created and maintained by the correlation mechanism, and that represent the current status of its deductions.
- *Alarm events* that are created from reasoning events and are sent to the interface for operator attention.
- *Internal error events* that are created when the correlation mechanism encounters an unexpected error.

Note**:** All of these events, except for the reasoning events, are sent and written to the TEC event database.

During installation and initial configuration, the Risk Manager `rmtec_cfg.sh` configuration shell program creates the Risk Manager event groups automatically. In addition to creating the event groups, this configuration program defines the filters for these event groups. For each event group that Risk Manager creates you must assign it to a Tivoli administrator. The following table summarizes the Risk Manager event group information for the four groups that are created by the `rmtec_cfg.sh` shell script.

Table 1. Risk Manager Event Groups

Event Container Name	Filter Defined	Event Group
RiskMgr_Authorized_Scanners	A_AuthorizedScan	Gets all events with class A_AuthorizedScan.
RiskMgr_Situations	A_Situation	A_Situation gets all events of class: - A_Situation1 - A_Situation2 - A_Situation3
RiskMgr_All	IDS_Event	Gets all events with class IDS_Event or subclassed from IDS_Event.

Event Container Name	Filter Defined	Event Group
RiskMgr_Exceptions	ComplexType	Gets all events of class C_Idsid
	InternalError	Gets all internal errors, such as: - IE_BadInput - IE_Situation - IE_TokenOverflow - IE_PrologFailure

After initial installation and configuration, you can modify the event groups, or create your own event group and define the filters for your newly created event group.

Figure 8 shows the event processing related to Risk Manager's event groups and event containers.

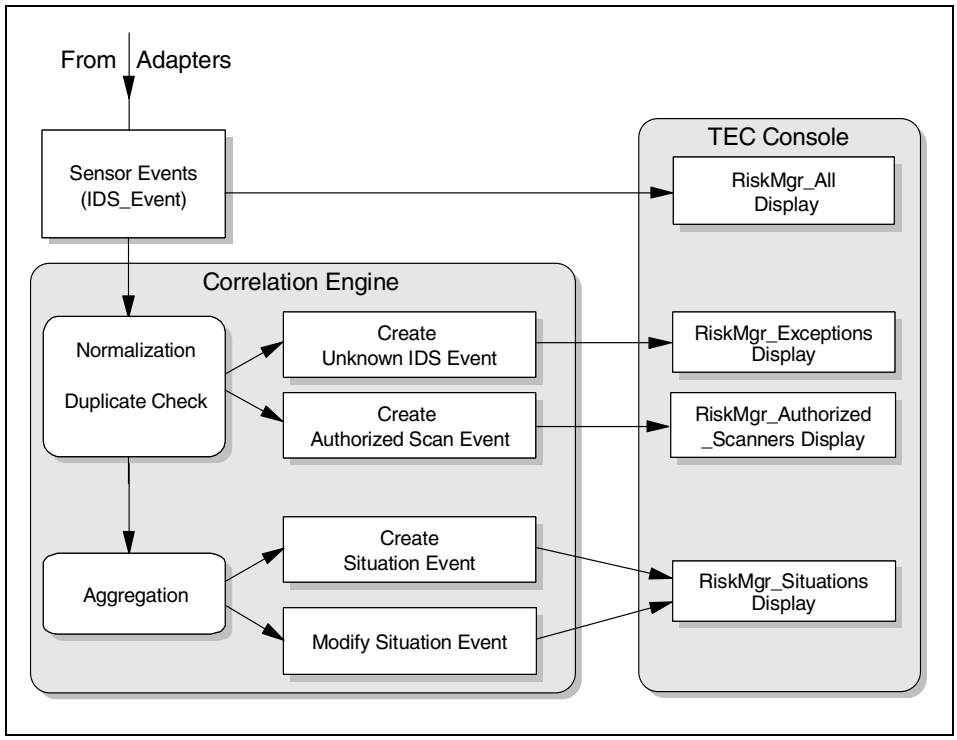


Figure 8. Event processing

4.2.4 BAROC files

Each adapter comes with a BAROC file that describes the classes of events that the TME adapter supports. This file is not used by the TME adapter itself, but it serves as a mandatory link between the adapter and the event server. The event server must load this file before it is able to understand events that are received from the adapter. A BAROC file usually has an extension of `.baroc`.

The Risk Manager BAROC files describe the TEC class hierarchy from which all the classes are derived from the EVENT class. The Risk Manager top-level and abstract classes, and the component TEC event class definitions can be found in these BAROC files (see Figure 7 on page 31, and the following figure).

Figure 9 on page 36 shows some event class definitions included in Risk Manager's `riskmgr.baroc` file.

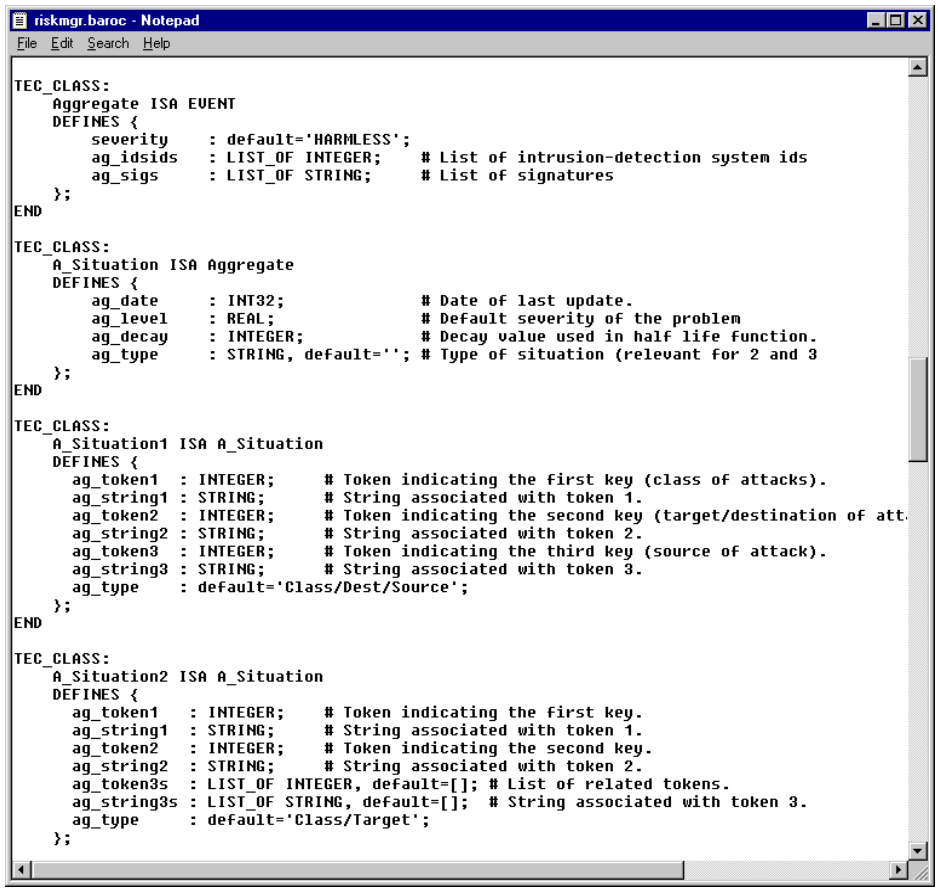


Figure 9. Risk Manager Baroc File

Table 2. Baroc Files

BAROC file name	Type of classes
riskmgr.baroc	The event classes. For event classes, a filter is defined and an event window (container) is created.
idwg.baroc	The top-level, abstract classes that are used to create the actual TEC event-class definitions. This file implements the generic data structures for alerts. The rules operate on these data structures. No instances of these classes should be sent to the TEC.
realsecure.baroc	The adapter for ISS RealSecure TEC event classes. Depends on idwg.baroc. This file contains the RealSecure class derivatives for the alert format.

BAROC file name	Type of classes
netranger.baroc	The adapter for Cisco Secure IDS TEC event classes. Depends on idwg.baroc. This file contains the Cisco Secure IDS (NetRanger) class derivatives for the alert format.
webids.baroc	Web Intrusion Detection event classes. Depends on idwg.baroc. This file contains Web Intrusion Detection class derivatives for the alert format.

4.2.5 Rules files

The rules files are compiled and loaded in a fixed order because of dependencies between them. The order in which the events are processed by the rules files corresponds to the order in which the rules files are loaded into the rule base. The control flow of the rules is dependent on that order, except when the Prolog flow control predicates are used (for example, commit action, commit rule, and commit set). See the following table for the exact order that the Risk Manager rules files (`.rls`) are loaded and processed:

Table 3. Risk Manager Rules Files

Rules File Name	Purpose of the Rules file
normalization.rls	This rules file contains statements to normalize the information related to: • The intrusion detection system that sent the event. Intrusion detection systems talking to the event console must be identified in the configuration of the console. If a new intrusion detection system sends messages, a severe alert is raised. • The destination identified in the attack. • The source identified in the alert. New sources are expected to occur every time, and they are generated automatically without information being displayed.
duplicates.rls	This rules file identifies alerts that are duplicates of the same attack, either because the information about the same attack is coming from multiple intrusion detection systems, or because a single attack triggers multiple alerts from the same intrusion detection system.
aggregates.rls	This rules file groups alert information in batches and processes them to see if there are obvious attack patterns and tool signatures.

Rules File Name	Purpose of the Rules file
boot.rls	This rules file contains statements that are executed when the Tivoli event server is restarted (for example, when a TEC Start event is received). There is also a section concerning testing to verify that the other rules files behave appropriately. This file is loaded last because the code contained is executed only in the case of the Tivoli event server startup, and it is very likely that the control flow has stopped the event processing before reaching this file.

Some aggregation rules included in `aggregate.rls` file are be seen in Figure 10 on page 39.

4.2.5.1 Rules template file

The riskmgr_templates.pro file contains Prolog predicates used by the rules files (`.rls`). This file handles the core of the correlation functions, including computation of the decay functions, maintenance of the situation facts, and generation of the alerts that appear on the user interface.

4.2.5.2 Rules configuration file

Risk Manager uses the rules configuration file (riskmgr_config.pro) to define host information for rules processing and to set thresholds related to the raising of alarms (see Chapter 4.5, “Managing TEC Correlation” on page 55 for more details). Risk Manager raises alarms when the Risk Manager event correlation processor encounters an event from an IDS host for the first time that is not defined in the rules configuration file. When TEC Correlation is installed and initially configured, Risk Manager sets default threshold values. As you use TEC Correlation, you might want to adjust these values if you are getting too many or too few alarms. You can customize threshold values by editing the riskmgr_config.pro configuration file (see Section 4.5.6, “Defining thresholds” on page 60). Whenever you edit this file, you must re-run the `rmtec_cfg.sh` shell script.

Figure 10. Risk Manager aggregation rules file

4.2.6 Situations

During the correlation phase intrusion detection events are brought into a relationship to situations. A *situation* is a set of findings that exploit the event flow in a particular context. Each situation is based on a view of the events in a specific direction.

Directions identified at this stage are the:

- Class (type of attack/signature)
- Target (destination of the attack)
- Source (source of the attack)

Additional information (such as the time and date of the attack, the severity, and the trust or confidence) is used to assess the situation. However, this information is not used as direction classifiers.

From this information, the following situation types are defined:

Table 4. Situation Classes

Type of Situation	What Is Evaluated	Goal	Trigger
Situation 1 "Class/Target/Source"	Evaluate individual triplets (signature, source, and destination).	This situation aims at detecting very serious single events, such as somebody obtaining a password file, or a login on a protected machine.	This violation is serious and requires immediate action.
Situation 2-1 "Target/Source"	For each source and destination, aggregate the type of attacks.	The goal of this situation is to monitor the data for patterns of attacks between two machines.	This situation is triggered if situation 1 is not triggered and the situation 2-1 criterion is met.
Situation 2-2 "Class/Target"	For each destination and signature, aggregate the source.	The goal of this situation is to monitor the data for patterns of attacks against a specific machine.	This situation is triggered if situation 1 is not triggered and the situation 2-2 criterion is met.

Type of Situation	What Is Evaluated	Goal	Trigger
Situation 2-3 "Class/Source"	For each source and signature, aggregate the destination.	The goal of this situation is to monitor the data for patterns of attacks from one machine to a list of targets.	This situation is triggered if situation 1 is not triggered and the situation 2-3 criterion is met.
Situation 3-1 "Source"	For each source, aggregate the destination and type of attack.	The goal of this situation is to monitor the data for widespread attacks.	This situation only reports information if situation 1 and all situation 2s are not triggered, and the situation 3-1 criterion is met.
Situation 3-2 "Target"	For each destination, aggregate the source and type of attack.	The goal of this situation is to monitor attacks against specific, high visibility hosts (for example, a Web server).	This situation reports information only if situation 1 and all situations 2s are not triggered, and the situation 3-2 criterion is met.
Situation 3-3 "Class"	For each signature, aggregate the source and destination.	The goal of this situation is to monitor for a particular kind of abuse, against a whole set of machines, coming from diverse attackers. This kind of situation reports attacks typically triggered by a post on BUGTRAQ, identifying a new vulnerability against a service that a large number of people are trying out.	This situation reports information if situation 1 and all situations 2s are not triggered, and the situation 3-3 criterion is met .

Triggering is the same for all situations. A situation has a sensitivity value and a threshold. The sensitivity value is computed using a mathematical formula from the previous value, the time since the last update, the sensitivity value of

the new event, and the confidence of the new event. When the sensitivity value exceeds the threshold, Risk Manager raises an alarm (see Chapter 4.5, “Managing TEC Correlation” on page 55 for related information).

4.3 TEC Correlation Engine

Let’s have a closer look at the correlation process.

4.3.1 Correlation process

The correlation process phases include:

- Normalization
- Duplicates
- Aggregation
- Cleanup

Correlation is handled using Prolog and rules files. The Prolog (`.pro`) and rules (`.rls`) files process TEC events, matching a set of event filters and using a Prolog engine.

Each of the correlation phases corresponds to a matching rules file (`.rls`, see the following figure). The Prolog files (`.pro`) contain definitions (*facts*) and functions (*predicate*s) called by the rules files.

4.3.1.1 Normalization phase

The normalization phase enables TEC Correlation to get standard information for every event.

The TEC Correlation’s `normalization.rls` file contains statements to normalize the information related to the:

- Date of the event (seconds since 1970)
- Intrusion detection system that sent the event
- Destination identified in the attack
- Source identified in the alert

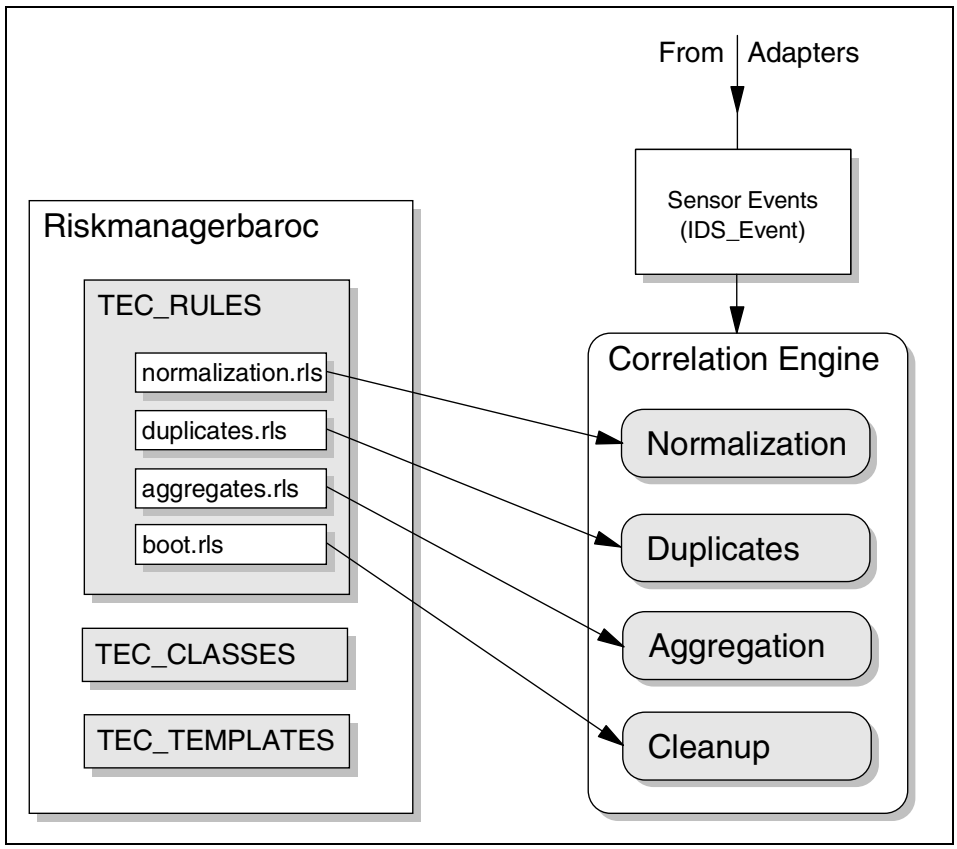


Figure 11. Correlation Process

Date of the event

The date contained in the event is transformed into a Prolog format and checked for consistency with the dates of recent events. When there is a time lapse of more than one month between the current date and the date of the last received event, an error message is raised on the console. Then the date of the last received event is reset to the date of the current event. Normal processing then applies. If the date cannot be processed because of an invalid date format or value (such as 0 or infinite), an error message is sent. In this case, the correlation engine leaves the event open on the TEC event console.

Intrusion detection system that sent the event

Intrusion detection systems are identified by a unique token, meaning unique in the IDS token space (an integer number between 0 to 99999). IDS tokens are defined in riskmgr_config.pro. If an IDS is not defined in the

riskmgr_config.pro file, a token is automatically assigned. At the same time, a message is sent to the interface to warn of the new IDS sending events. If the new IDS event cannot be created, an error message is sent to the TEC event console that indicates the likely cause of the problem. The correlation engine stops processing the event, and the event remains open. The warning can be turned off using the `ignore_ids_creation` parameter in riskmgr_config.pro. Note that the automatically assigned tokens are considered volatile because they disappear in the event of a TEC shutdown.

Destination identified in the attack

Attack destinations (or targets) are identified by a unique token, meaning unique in the destination token space (an integer number between 0 to 99999). Tokens can be defined in the riskmgr_config.pro file using the `destination` parameter. If a destination is not defined in the riskmgr_config.pro file, a token is automatically assigned to it and available values in the event are used. If there is a problem processing the destination information and assigning the token, an error message indicates the cause of the problem, the correlation engine stops processing the event, and the event remains open. Note that the automatically assigned tokens are considered volatile because they disappear in the event of a TEC shutdown.

Source identified in the alert

Attack sources are identified by a unique token, meaning unique in the source token space (an integer number between 0 to 99999). Tokens can be defined in the riskmgr_config.pro file using the `source` parameter. If a source is not defined in the riskmgr_config.pro file, a token is automatically assigned to it and available values in the event are used to fill in the predicate. In case of a problem with processing the source information and assigning the token, an error message indicates the cause of the problem, the correlation engine stops processing the event, and the event remains open. Note that the automatically assigned tokens are considered volatile because they disappear in the event of a TEC shutdown.

Source tokens are also used to handle authorized scans. Some machines can be explicitly authorized to be sources of alerts. To define them, a source value must be defined with the `source` definition. Then, an `authorized_scan` parameter is configured with the token related to this source in the riskmgr_config.pro. All events originating from this source are regrouped in an A_AuthorizedScan container. When the alert does not contain appropriate information, an internal error message is issued by TEC Correlation. Only four internal error messages are issued:

- IE_BadInput
- IE_TokenOverflow

- IE_PrologFailure
- IE_Situation

4.3.1.2 Duplicates phase

The duplicates phase enables Risk Manager TEC Correlation to determine whether an intrusion detection event has already been received and processed. Then events are weighed in relation to one another.

The `duplicates.rls` rules file identifies alerts that are duplicates of the same attack.

This phase determines how duplicate events are caused. A single alarm might represent a large number of duplicate events. Either the same event has been updated, or multiple alerts have been sent for the same attack by the same IDS tool. Or, two different IDS tools might recognize the same attack and both send alerts. If no duplicates are found, correlation goes on to the aggregation phase.

4.3.1.3 Aggregation phase

After the external event has been filtered for duplicates, the aggregation phase enables TEC Correlation to aggregate the external event to the appropriate internal events. The aggregation mechanism is based on the similarity of events, the correlation is based on timing and entities involved, and the escalation policy is based on the number of events received, whether these events are successful or not, and additional information provided by the configuration of the system.

The `aggregates.rls` rules file groups alert information in batches and processes them to look for obvious attack patterns and tool signatures.

A single alarm might represent an aggregation of events, based on a variety of situations, that are detected by the correlation rules. The aggregate events include (see Figure 7 on page 31):

A_Situation1

A_Situation2

A_Situation3

A_AuthorizedScan

Situation1 events are referred to as a *triplet* - meaning class, target, and source in that order. Class is a group of signatures (or list of strings). Target is

the event destination. Source is the event source (see Figure 12 and Chapter 4.4.1, “Monitoring correlation examples” on page 49 for additional information).

Situation2 events are referred to as a *tuple*. There are three possible combinations:

- class/target
- class/source
- target/source

For Situation2 events, a larger number of events is grouped together to be more sensitive to attack patterns and tool signatures. Situation2 aggregate events might exceed the thresholds even though Situation1 might not have.

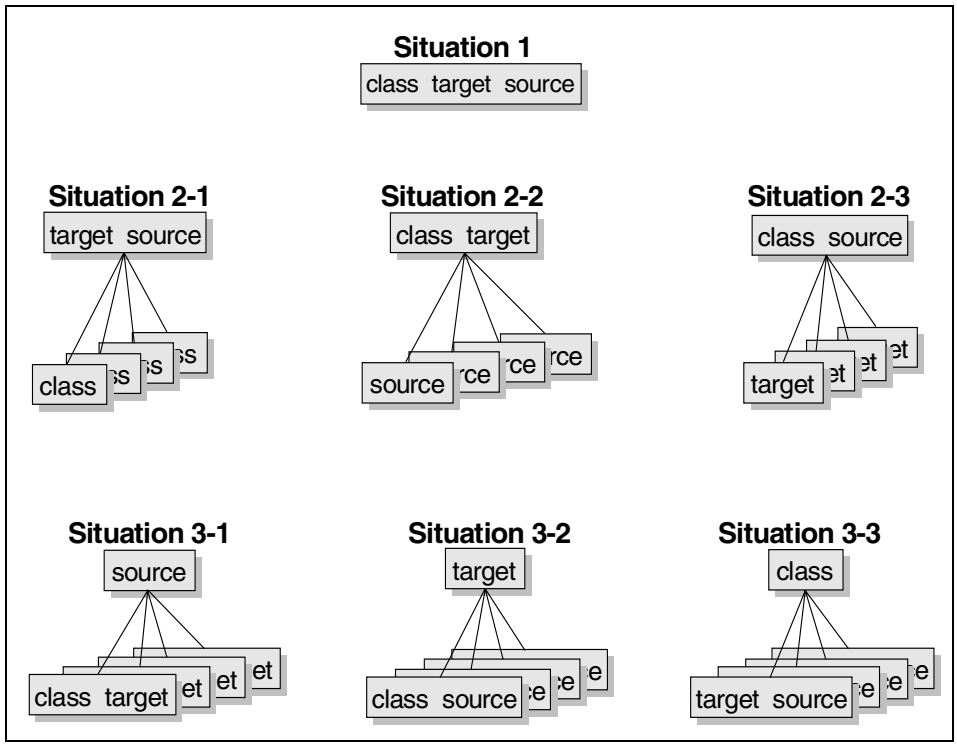


Figure 12. Situations

For *Situation3 events*, even larger sets of events are grouped together. Situation3 events are grouped together to see if anything shows up in the pattern, depending on the thresholds in the rules file:

- class [targets] [sources]
- target [classes][sources]
- source [classes][targets]

Aggregation events are basically the outcome of the correlation. These events appear at the event console. The aggregate phase accumulates a list of signatures together and then, using the decay function, issues a warning or an alert if the thresholds in the rules file are exceeded. Warnings appear at the event console as closed events. They are generally less precise, less severe, duplicates, and so forth. You cannot delete events; you can only close them. Alerts are reported only if they reach certain thresholds. An alert remains open until you, as the Tivoli administrator, analyze the alert details and handle it appropriately. After handling occurs, the alert should be closed.

A decay function is used in the aggregate phase to decrease the importance of events with the passage of time. The decay value is a half-life, so a value of 600 seconds results in events being decreased in importance by a factor of 2 after 10 minutes. Decay values of 600 and 3600 are used, but this is not configurable by the end user.

4.3.1.4 Clean-up Phase

The clean-up phase enables TEC Correlation to perform routine maintenance tasks, such as cleaning up the memory. Clean-up is done periodically to make sure that certain files do not grow too large. The clean-up procedure also refreshes the information available at the event console every five minutes.

The `boot.rls` rules file contains statements that are executed when the Tivoli event server is restarted (for example, when a TEC Start event is received).

4.4 Monitoring Risk Manager intrusion detection events

When an intrusion detection sensor detects suspicious activity, it collects the data and forwards the data to the event console. Risk Manager either logs the alarm event or generates an alarm. The alarm is forwarded to the event console where the Tivoli administrator is monitoring intrusion detection events. The Tivoli administrator, by monitoring events on the event console, can take the feedback into account when specifying a lesser or higher value than the one originally given by the sensor. This way, the operator can reconfigure the sensors to discard known probes (such as an internal security scan) or adjust differences between tools. For example, the administrator can adjust the difference in the trust levels between e-mail alarms given by the ISS RealSecure and Cisco Secure IDS products. For Web Intrusion

Detection, the administrator can add or remove trusted hosts. The administrator monitors these types of Risk Manager intrusion detection events:

- RiskMgr_All
- RiskMgr_Situations
- RiskMgr_Authorized_Scanners
- RiskMgr_Exceptions

Risk Manager classifies events to indicate their degree of severity. The predefined severity levels, in order of decreasing severity, are:

- FATAL
- CRITICAL
- MINOR
- WARNING
- HARMLESS
- UNKNOWN

Risk Manager changes all intrusion detection events with a severity of MINOR, WARNING, and HARMLESS to a status of CLOSED. Risk Manager only shows a status of OPEN for intrusion detection events with a severity of FATAL, CRITICAL, or UNKNOWN. All events related to situations and created by Risk Manager during the correlation process show an OPEN status.

Next to each Risk Manager event group container, you see a severity indicator icon when any events are stored in the container. This icon indicates the severity of the events currently in the container. As the severity level changes, the icon changes to match the severity level.

Figure 13. Risk Manager Event Groups

4.4.1 Monitoring correlation examples

In this chapter we describe three examples of different situations and how the related information can be monitored using the TEC console.

Figure 14 and Figure 15 on page 51 show a Situation1 event on the Risk Manager's RiskMgr_Situations event screen (see Section 4.4, "Monitoring Risk Manager intrusion detection events" on page 47) and the related detailed view of the event. Situation1 evaluates individual triplets - class (signature), destination (target), source - (see Table 4 on page 40). This situation aims at detecting very serious single events.

In the example, the Situation1 event was generated because of a Denial of Service attack from a single host (source, IP address - 9.3.187.129) against another single host (destination/target, IP address - 9.3.187.154).

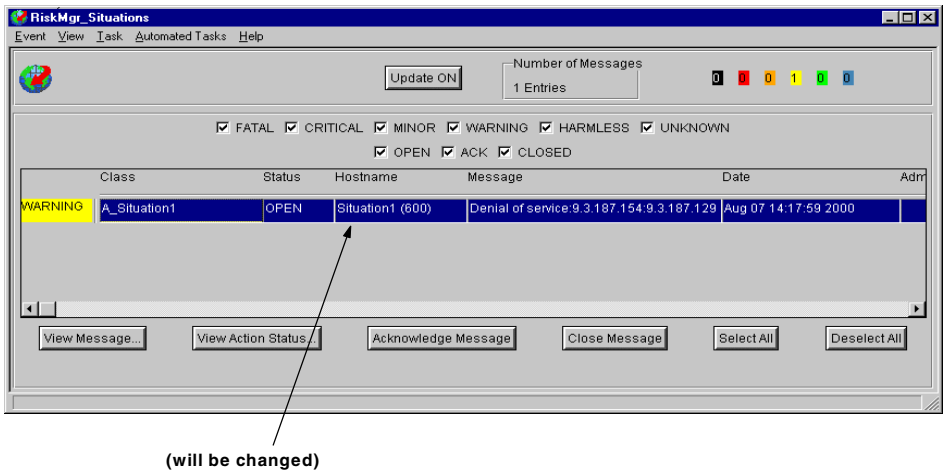


Figure 14. Example: Situation 1 (Class/Target/Source)

The following Situation1 information is included in Figure 15:

- Class: 'Denial of service' (signature generated by TEC Correlation)
- Target: IP address is 9.3.187.154.
- Source: IP address is 9.3.187.129.

The ag_sigs field contains signatures of several attacks as given by the IDS:

- 'Repeated Arp (500)'
- 'Repeated Arp (200)'
- 'Repeated Arp (100)'

- ‘Repeated Arp (50)’
- ‘Repeated Arp (10)’
- Arp

A decay function is used to decrease the importance of events with the passage of time. The decay value is displayed in the ag_decay field (600, that means 600 seconds) and along with the situation type in the `hostname` field (‘Situation1 (600)’). The decay value is a half-life, so a value of 600 seconds results in events being decreased in importance by a factor of 2 after 10 minutes. Decay values of 600 and 3600 are used, but this is not configurable by the end user.

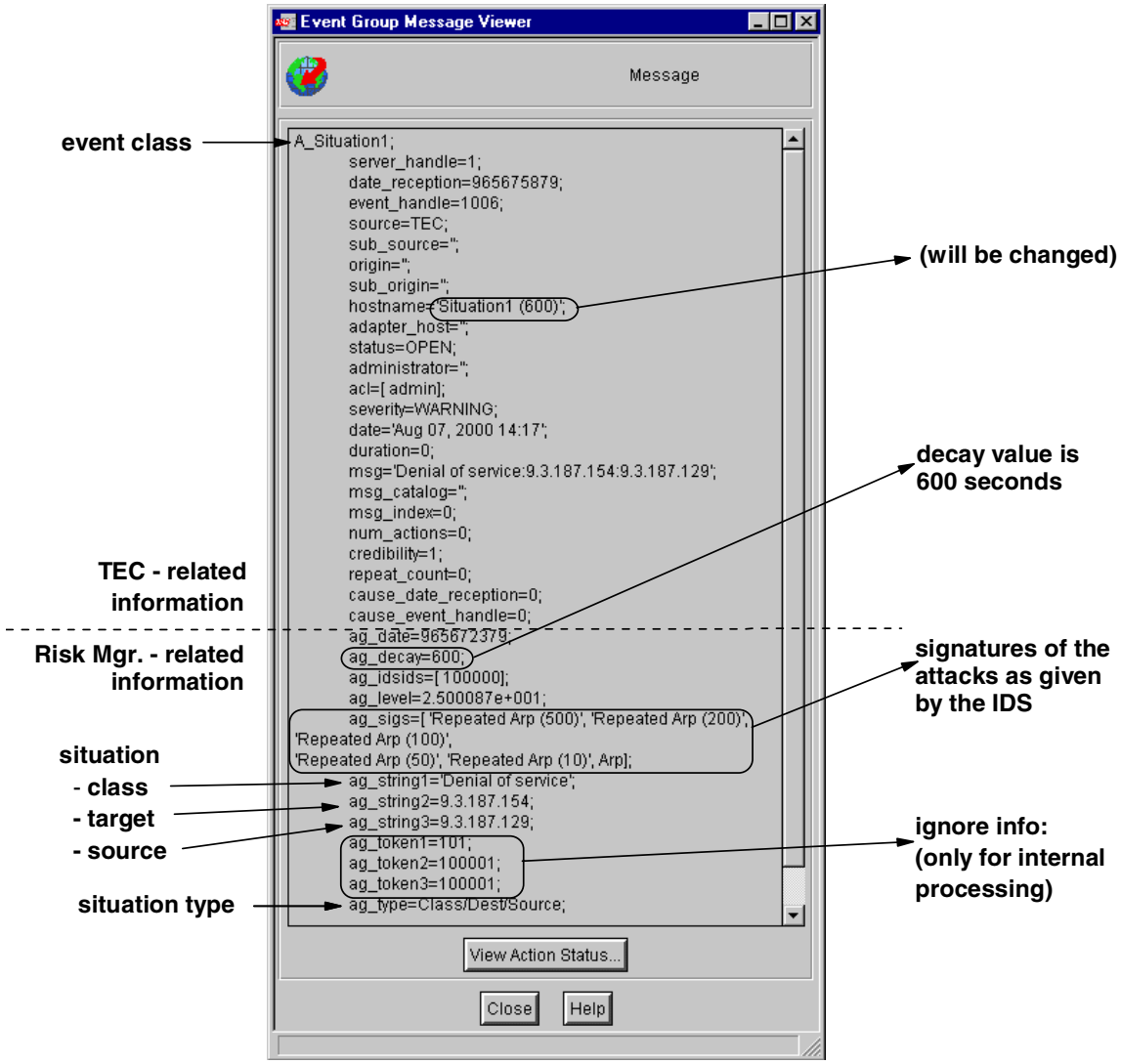


Figure 15. Example: Situation 1 (Class/Target/Source), Event View

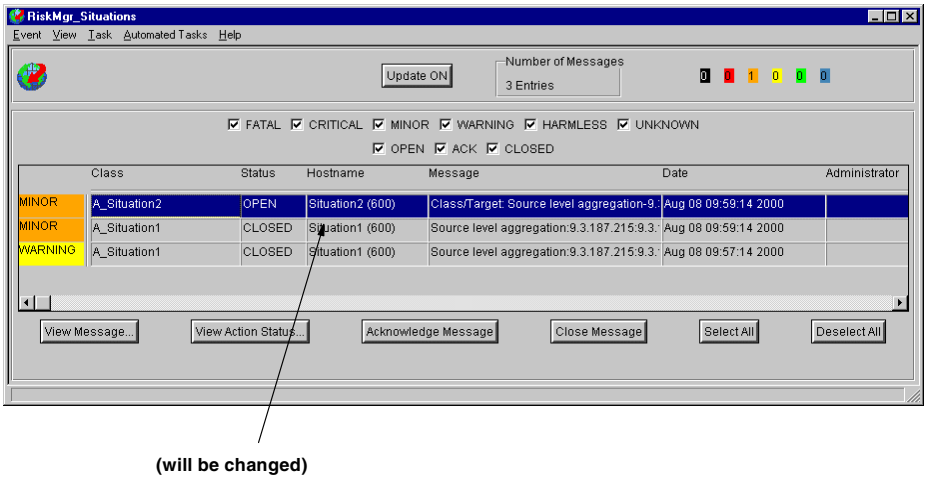


Figure 16. Example: Situation 2-2 (Class/Destination)

The next example (Figure 16, and Figure 17 on page 53) can be described as a Distributed Denial of Service attack. Related to it is a Situation 2-2 (class/target) event. This event type is generated when multiple host machines (source) attack a single target machine (destination/target).

The following Situation2 information is included in Figure 17:

- Class: ‘Source level aggregation’ (signature generated by TEC Correlation)
- Target: IP address is 9.3.187.215.
- Source: IP addresses:
  - 9.3.1.187.
  - 9.3.240.135.
  - 9.3.187.220.
  - 9.3.187.221.
  - 9.3.187.224.

The ag_sigs field contains signatures of several attacks, started from five different systems against one single target host as given by the IDS:

- ‘Repeated PingFlood (100)’
- ‘Repeated PingFlood (50)’
- ‘Repeated PingFlood (10)’

- PingFlood
- Netbios_Session_Request
- Windows_Access_Error

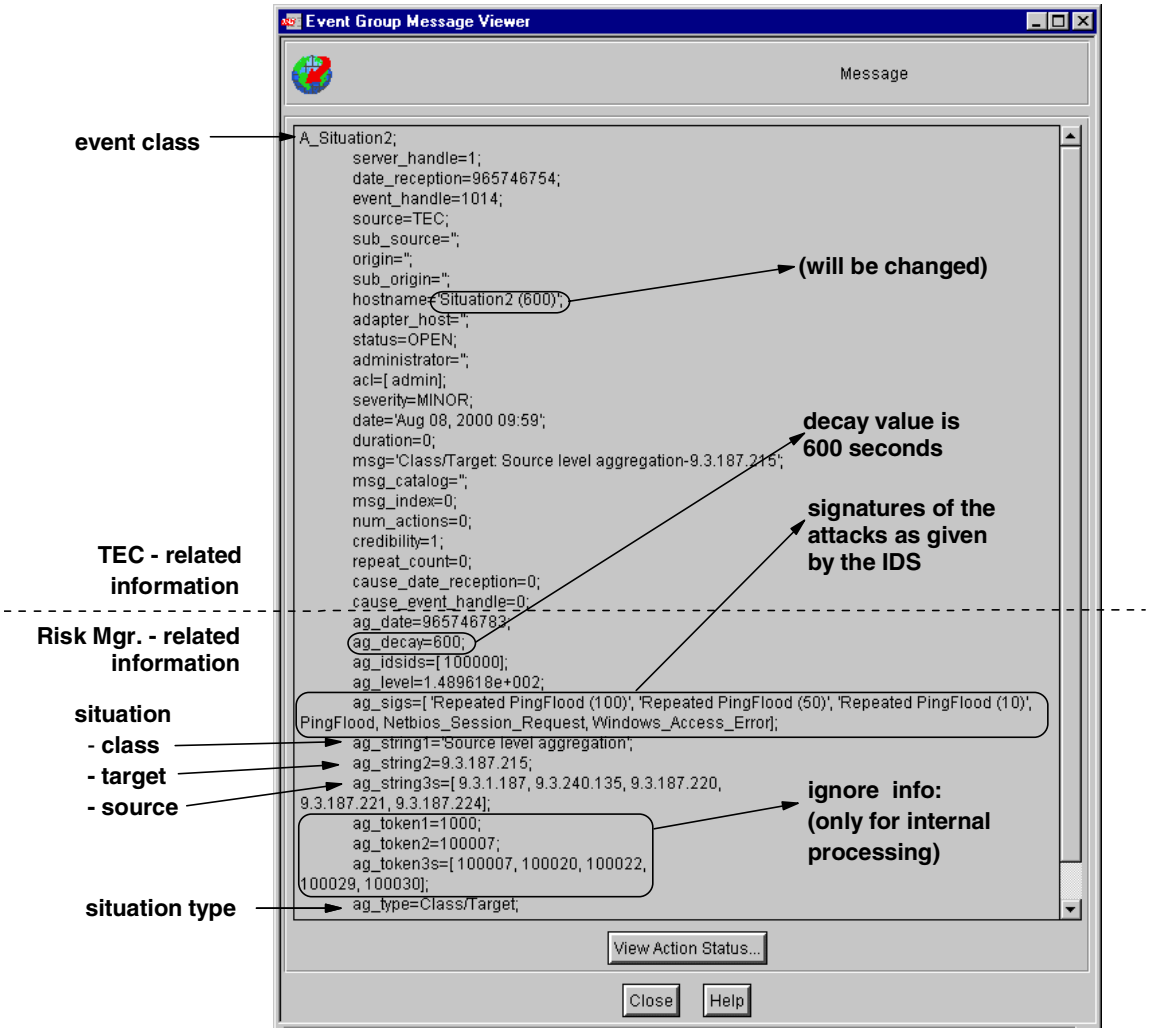


Figure 17. Example: Situation 2-2 (Class/Destination), Event View

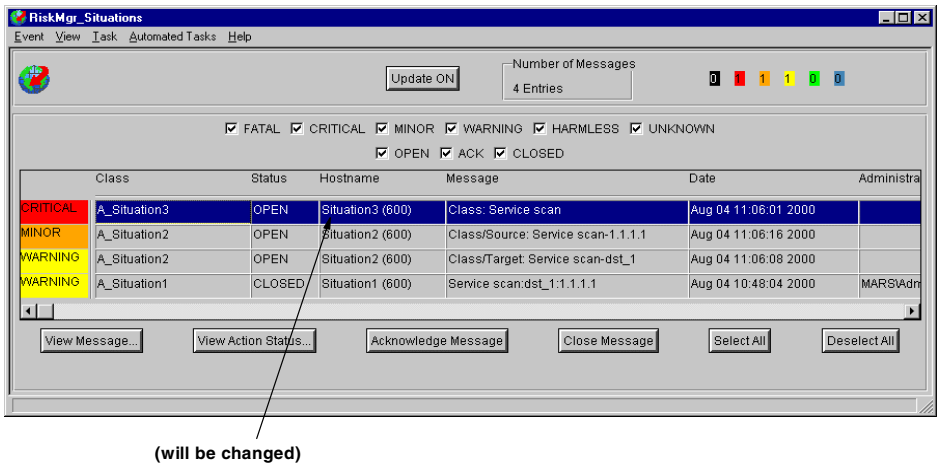


Figure 18. Example: Situation 3-3 (Class)

The third example of situations describes a Service Scan. Related to this example is a Situation 3-3 event (class) and generated when multiple host machines (source) attack multiple target machines (source/target) using a single type of attack (signature).

The following Situation3 information is included in Figure 18 and Figure 19 on page 55:

- class: 'Service scan' (signature generated by TEC Correlation)
- target: host names
  - dst_1
  - dst_2
  - dst_3
- source: IP addresses
  - 1.1.1.1.
  - 1.1.1.2.
  - 1.1.1.3.

The ag_sigs field contains the signature of an attack as given by the IDS:

- 'Access attempt to service_1'

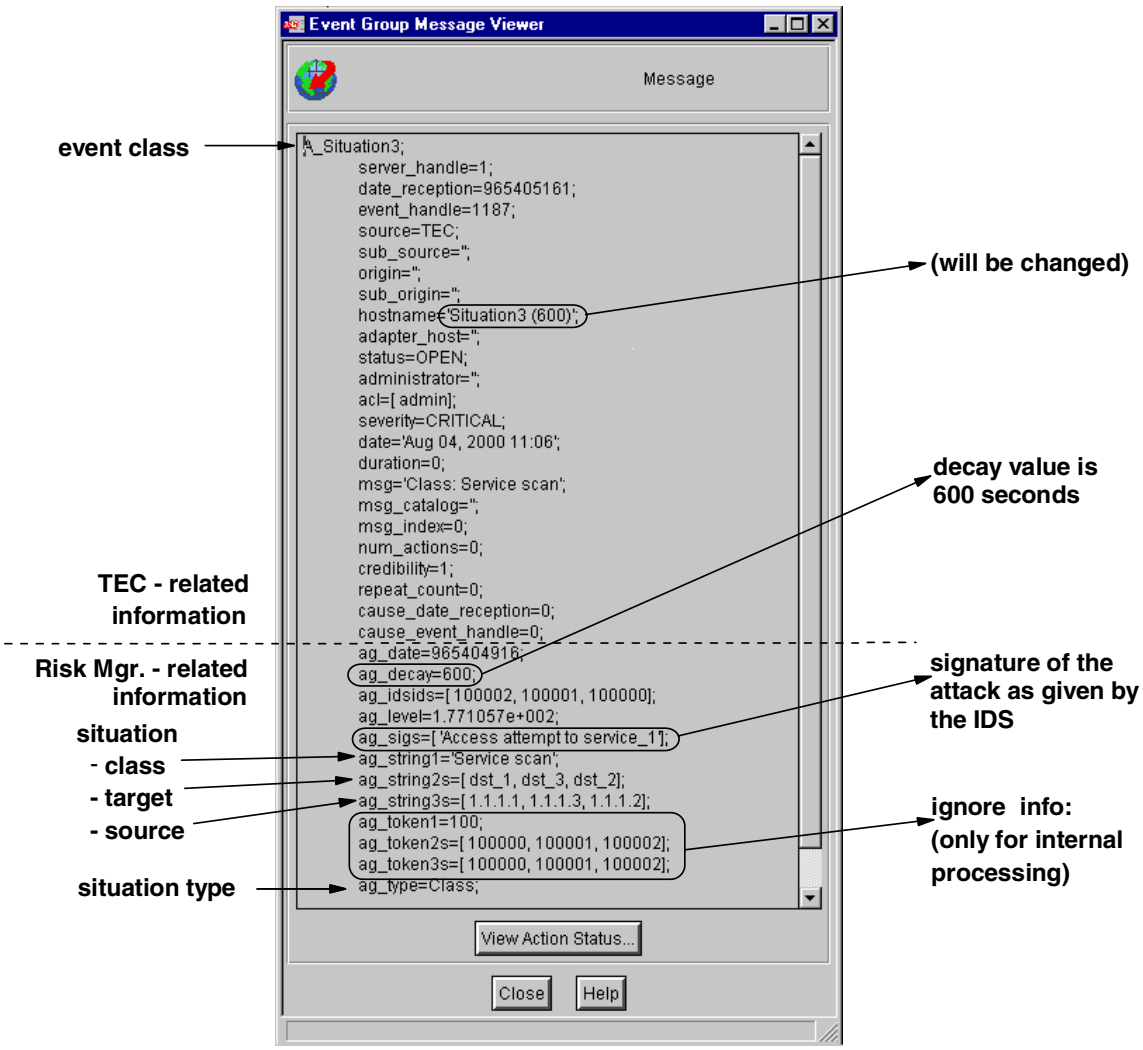


Figure 19. Example: Situation 3-3 (Class), Event View

4.5 Managing TEC Correlation

You can configure the TEC Correlation using the riskmgr_config.pro configuration file. This file is used to define host information for rules processing and to set thresholds related to the raising of alarms. Whenever you edit this file, you must re-run the `rmtec_cfg.sh` file.

The configuration file contains these customizable definitions:

- Known IDS
- Ignore IDS settings
- Destination host
- Source host
- Authorized scan host
- Threshold

Other management tasks include monitoring Risk Manager intrusion detection events and handling TEC Correlation errors.

4.5.1 Defining known intrusion detection systems

You can define a set of intrusion detection systems that are installed, running, and generating event information. When an event is received at the Tivoli event server from any IDS system that is not already defined in this list, an alarm is generated ti inform the operator of the new system.

To define an intrusion detection system, type:

```
ids(arga,argb,argc,argd,arge ).
```

The period (**.**) is required.

Where:

arga The IDS token value (integer number). Valid range of numbers is 0 to 99999.

argb The intrusion detection tool name (string).

argc The host name where the intrusion detection system is located (string).

argd The IP address of the IDS tool (string).

arge This argument must be an underscore.

For example:

```
ids(10,'tivoliids','mymachine','1.1.111.11',_).
ids(11,'otherids','mymachine2','2.2.222.22',_).
```

The following figure shows the definition of the ISS RealSecure intrusion detection system and the ignore definition of the Klaxon intrusion detection system.

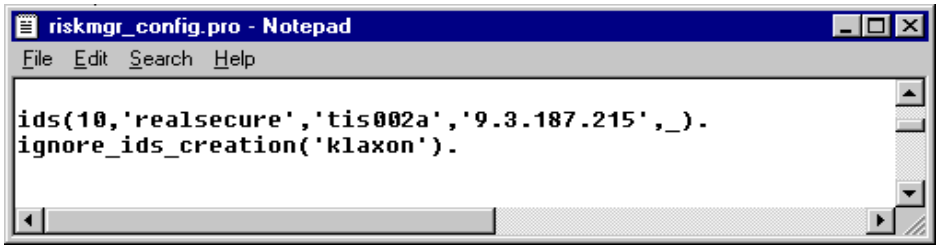


Figure 20. Defining/Ignoring intrusion detection systems

4.5.2 Ignoring an intrusion detection system setting

You can hide automatic IDS creation, meaning that you can ignore discovery for known intrusion detection systems of a certain type. The default settings for automatic IDS creation include these intrusion detection systems:

- `realsecure`
- `netranger`
- `webids`

For example, if you have Windows NT running at the endpoint, you might want to ignore the Cisco Secure IDS (NetRanger) intrusion detection system.

```
ignore_ids_creation('netranger').
```

The period (**.**) is required.

Likewise, if you have Solaris running at the endpoint, you might want to ignore the ISS RealSecure intrusion detection system (realsecure).

4.5.3 Defining destination hosts

You can define a set of hosts that can be potential targets for attack. Defining destinations, in particular those with multiple names, ensures that an appropriate name is displayed on the event console.

When a server has several names or several network interfaces, you can define multiple destinations with one token (that means assign a number between 0 to 99999). In this case, the name or IP address appearing on the event console is the first name or IP address defined in the token.

You should use the same token for the destination and the source when a host is designated as both. See “Defining Source Hosts” for information on defining the source host.

To define a new destination:

```
destination(arga,argb,argc ).
```

The period (.) is required.

Where:

arga The destination token value (integer number). Valid range of numbers is 0 to 99999.

argb The host name of the machine protected by one or more IDS tools (string).

argc The IP address of the destination server.

For example:

```
destination(100,'mymachine','1.1.111.11').
destination(100,'mymachine','10.10.10.11').
destination(101,'mymachine2','2.2.222.22').
destination(101,'mymachine2.companya.com','2.2.222.22').
destination(101,'mymachine2','2.2.222.122'.
destination(101,'mymachine2.companya.com','2.2.222.122').
destination(101,'machine2alias','22.22.22.12').
destination(101,'machine2alias.companyalias.com','22.22.22.12').
destination(102,'mymachine3','1.2.232.99').
destination(102,'mymachine3','1.2.232.199').
destination(102,'mymachine3','11.11.11.99').
```

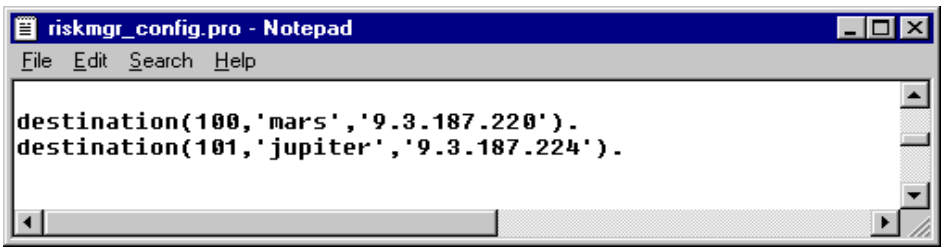


Figure 21. Defining destination hosts

4.5.4 Defining source hosts

You can define source entries that identify the machine information for attack sources. The primary purpose for defining sources is to permit the definition of *trusted* sources that might periodically run simulated attacks against systems in the network.

You can define multiple sources using a single token because a server might have several names or several network interfaces. In this case, the name or the IP address that appears on the event console is the first name or IP address defined.

You should use the same token for destination and source definitions when a host is designated as both. Also see Section 4.5.3, “Defining destination hosts” on page 57.

To define a source, type:

```
source(arga,argb,argc ).
```

The period (**.**) is required.

Where:

arga The source token value (integer number). Valid range of numbers is 0 to 99999.

argb The host name of the source machine (string).

argc The IP address of the source machine (string).

For example:

```
source(1000,'mymachine2','2.2.222.22').
source(1000,'mymachine2','2.2.222.123').
source(1000,'mymachine2','22.22.22.12').
source(1001,'mymachine3','1.2.232.99').
source(1001,'mymachine3','1.2.232.199').
source(1001,'mymachine3','11.11.11.99').
```

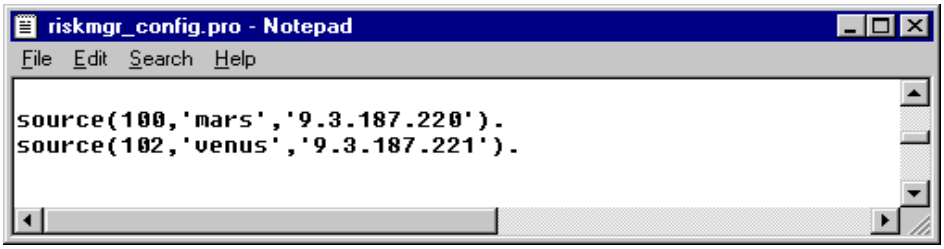


Figure 22. Defining source hosts

4.5.5 Defining authorized scanners

You can define source hosts that are authorized to launch scans that might be interpreted as attacks. Hosts that are defined as authorized scanners must also be defined as source hosts. Only set trusted hosts as authorized scanners as an actual intrusion from that host will not generate an alarm.

To define an authorized scanner, type:

```
authorized_scan(arga ).
```

The period (**.**) is required.

Where:

arga The source hosts that are authorized to launch scans. The source value (defined source token) must correspond to the appropriate source entry.

For example:

```
authorized_scan(1000).
authorized_scan(1001).
```

4.5.6 Defining thresholds

You can define thresholds that control when a series of events is escalated to alarm status.

To define thresholds, type:

```
threshold(arga,argb,argc,argd,arge,argf,argg,argh ).
```

The period (**.**) is required.

Where:

arga The situation name. This name must be one of the following:

- 'situation1'
- 'situation2'
- 'situation3'

argb The type of situation:

- For 'situation1' this must be an underscore (_)
- For 'situation2' this must be underscore (_), 'Class/Target', 'Class/Source', or 'Target/Source'

- For ‘situation3’ this must be underscore (_), ‘Class’, ‘Target’, or ‘Source’

The single quotes (‘’) are mandatory.

argc The token for classes of attack can take the following values defined in Table 5:

Table 5. Tokens for Classes of Attack

Token Value	Class of Attack
1	Web-related alerts
2	Network management attacks
3	E-mail anomalies
4	User commands
5	Targeted denial of service
6	Service compromise attempt
7	Trojan horse traffic
100	Service scan
101	Denial of service
1000	Source-level aggregation
10000	Target-level aggregation
100000	Class-level aggregation

argd See Table 6 on page 62.

arge See Table 6.

argf The ‘WARNING’ threshold (a positive real number). When the level of the situation exceeds this value, an event of WARNING severity is issued.

argg The ‘MINOR’ threshold (a positive real number). When the level of the situation exceeds this value, an event of MINOR severity is issued.

argh The ‘CRITICAL’ threshold (a positive real number). When the level of the situation exceeds this value, an event of CRITICAL severity is issued.

Table 6. Token Definitions

Situation	Type	Token *argc*	Token *argd*	Token *arge*
'situation1'	—	Class of attack	Destination of attack	Source of attack
'situation2'	—	Class or destination of attack	Destination or source of attack	An underscore (_)
	'Class/Target'	Class of attack	Destination of attack	An underscore (_)
	'Class/Source'	Class of attack	Source of attack	An underscore (_)
	'Target/Source'	Destination of attack	Source of attack	An underscore (_)
'situation3'	—	Class or destination or source of attack	An underscore (_)	An underscore (_)
	'Class'	Class of attack	An underscore (_)	An underscore (_)
	'Target'	Destination of attack	An underscore (_)	An underscore (_)
	'Source'	Source of attack	An underscore (_)	An underscore (_)

Examples:

```
threshold('situation1',_,1,_,_,10,100,500).
```

->I'm only interested in large Web attacks (see Figure 23 on page 63).

```
threshold('situation1',_,_,10,_,5,10,15).
```

->If destination host 10 is touched, I want to know it immediately.

Note that for this to work, at least one 'destination' parameter defining token 10 must exist.

```
threshold('situation2','Class/Target',_,10,_,5,10,15).
```

->The same rule concerning host 10 applies for situation2.

```
threshold('situation3','Source',50,_,_,100,1000,10000).
```

->I want to know as little as possible about attacks coming from the source associated with token 50.

Note that for this to work, at least one 'source' fact defining token 50 must exist (see Figure 23).

```
threshold('situation1',_,_,_,_,0.2,0.4,0.6).
```

->I want to escalate events extremely quickly (not advisable).

Irrelevant tokens:

- For ‘situation1’ the *argb* parameter is irrelevant.
- For ‘situation2’ the *arge* parameter is irrelevant.
- For ‘situation3’ the *argd* and *arge* parameters are irrelevant.
- The underscore (_) must be used for irrelevant tokens.

The three general `threshold` definitions must be preserved in this file. Only the actual threshold values can be adjusted: the ‘WARNING’ threshold, the ‘MINOR’ threshold, and the ‘CRITICAL’ threshold. The values provided should be acceptable in most cases:

```
threshold('situation1',_,_,_,_20,80,150).
threshold('situation2',_,_,_,_20,80,150).
threshold('situation3',_,_,_,_20,80,150).
```

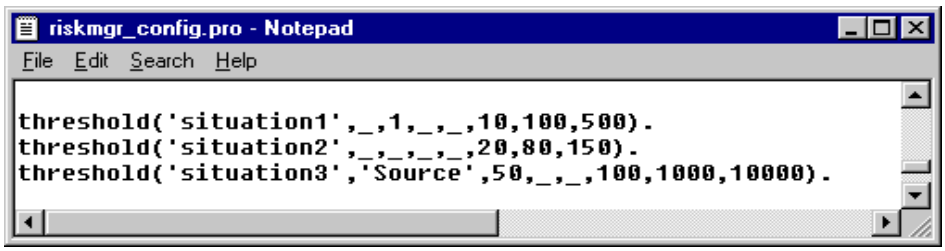


Figure 23. Defining threshold values

When situations appear with an inappropriate severity, the thresholds can be increased or lowered to fine tune them. All thresholds are positive real numbers, with *argf* being the smallest and *argh* being the largest.

4.6 TEC correlation errors

There are four internal error messages provided for TEC correlation. These messages include:

IE_BadInput
You receive this error message when the event message cannot be read correctly by the event console.

IE_Situation
You receive this error message when the application of the correlation algorithm failed on the event.

IE_TokenOverflow
You receive this error message when there is a token allocation failure, such as when TEC Correlation runs out of available tokens.

IE_PrologFailure
You receive this error message when there is a Prolog failure. When this happens, you must determine which file, which version of the file, or which predicate has caused the failure. This information is supplied in the event.

Chapter 5. Network Services Auditor (NSA)

In addition to the Risk Manager core product, which deals with the events originating from detectors deployed throughout the network, there is another component available for use. It can be downloaded at no additional charge by registered licensees of Tivoli SecureWay Risk Manager who wish to assess the vulnerability of their infrastructure in selectable depth, ranging from basic port scanning of a single machine to rigid examination of ports, servers, services, and known security vulnerabilities of all machines on a whole range of subnets. This tool is called Network Services Auditor (NSA), it can be run completely independent from Risk Manager to serve one or more of the following purposes:

- Do an initial scan of either the most exposed or the most critical parts of your network in support of security measures such as the deployment of an Enterprise Risk Management solution
- Measure the success of changes that you have applied
- Check for exposures on a regular basis, prepare for ethical hacking
- Carry out security audits
- Verify the compliance with corporate or unit security policies
- Produce useful and comprehensive reports on all of the above

This chapter will introduce the concept of NSA, discuss the main components and how they relate to each other, summarize the details that are essential to understand and use the tool, and discuss the context in which it can be used. It is not intended to replace the official User's Guide that comes with NSA. This document is part of the installation package, and is available in three different formats; HTML, PDF, and Postscript.

5.1 Introduction

In this section we will provide a brief summary of the capabilities of NSA and list the system platforms it can run on. We will also give an overview of the components and features.

5.1.1 Overview

Figure 24 on page 67 provides an overview of the steps and components involved with NSA. NSA's features and capabilities are:

- NSA imitates the first steps of an intruder who wants to gather certain information about a set of computer and networking systems. This is called *scanning*. For this the tool is called a scanner. The gathered information could then be maliciously exploited to either launch intrusion attempts (to gain unauthorized access to information stored on those systems), or to compromise these systems and cause them to fail.
- NSA documents its findings in any desired level of detail and in a variety of formats. This is called *reporting.* There is a rich set of options available to format and enhance the reports. By using these options, a report can be turned into a comprehensive document describing the security state of a network.
- In order to enable the separation of the above two steps, NSA uses its own database for intermediate storage of the scan results and for documentation of the scans themselves. This enables multiple reports based on the same scan, even a long time after completion of the scan.
- NSA allows users to define security policies in files. Policy definition files allow a site to define what servers and services they allow, as well as how severe a violation is. Policies can then be checked during the report phase.
- Almost every aspect of what NSA is doing is highly customizable. This includes the scan target systems, the scope of the scan on each of them, the behavior during the scan, the matching patterns that are considered as findings, the amount of scanning activity, and more.
- Report data can be made available to other applications by using the so-called *raw report*. This can be parsed and converted to the format required by that application.
- Additional code can be written by the user and added to NSA if the need arises. This is done by using the *NSA scripting language*, which has a syntax similar to C and PERL.

NSA has a staggering number of options. However, only a couple are needed in ordinary circumstances. While one may never need most of the options available, their presence provides NSA the flexibility to meet unusual situations that may arise.

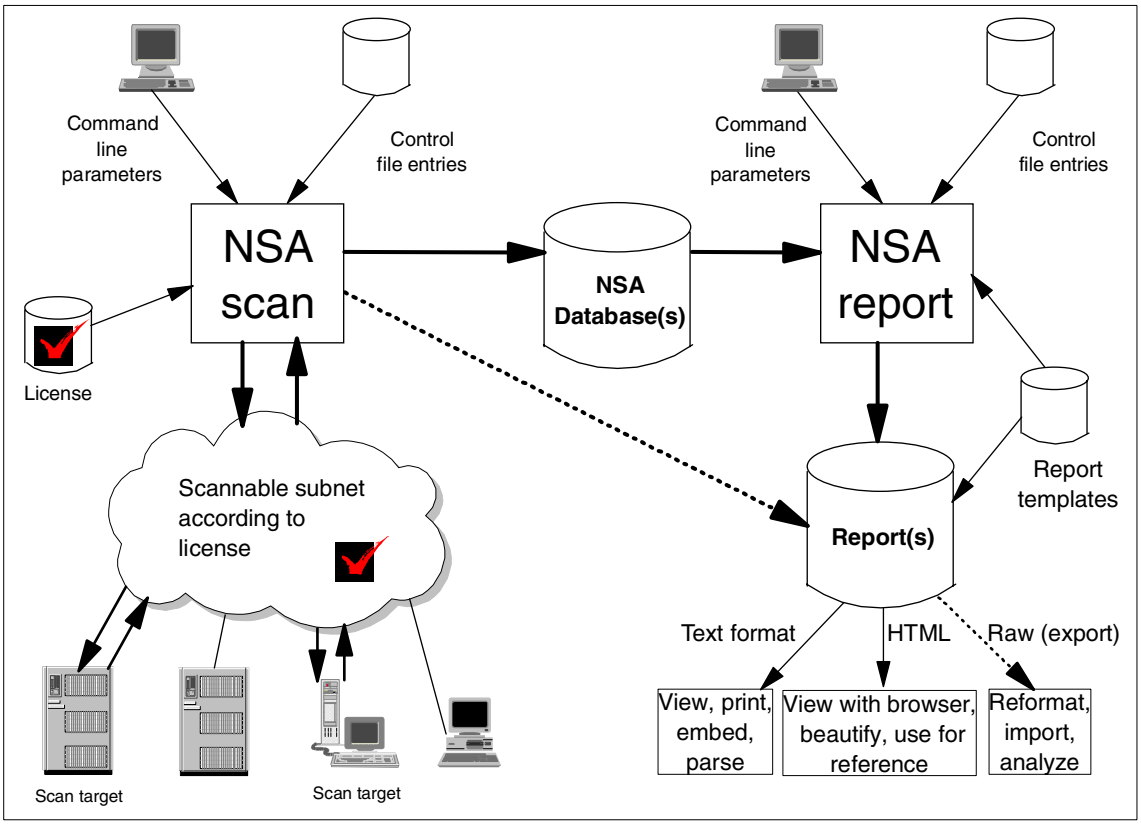


Figure 24. NSA functional overview

5.1.2 Supported platforms, components, capacity planning

The following can be useful for planning of the system and should be considered before the tool is being downloaded.

Platforms

NSA runs on UNIX based systems. In combination with Risk Manager it is currently available for the following platforms:

- AIX - 4.x
- Linux - libc6 based systems
- Solaris - 2.5.1 Sparc, 2.6 Sparc/x86, 7 Sparc

In general, versions compiled on earlier OS releases should work on later releases of the OS. In case of doubt, refer to the latest information on the download Web page. For Linux there used to be special considerations that

required testing with individual versions. With libc6 this is no longer the case. NSA has been tested with RedHat 4.x, 5.x, 6.x, S.u.S.E. 5.x, but others (including later versions of RedHat and SuSE) are more than likely to work.

Components

NSA is built of several parts. With the Risk Manager version they will all be installed as one package.

Main executable	Single file, does both scanning and reporting.
Support files	This includes configuration, policies, hosts, report related data, documentation, and more. The presence of these files is required at execution time when the corresponding function is being executed.
Utilities	Additional functions that have been separated from the main module
Scripting system	A fully functional scripting language. It turns NSA into a modular, easily expandable tool. This is intended to become the base of additions like vulnerability testing in the future. Short response times for new attack patterns can be expected. Near term extensions of this feature can be expected.

Capacity planning

Although there are no strict hardware requirements, the following are some general rules. The most important requirement for the system that hosts NSA is sufficient network bandwidth. Main memory is second, and CPU speed third. NSA can be run on portable systems such as an IBM Thinkpad with Linux. This enables you to use it with different subnets in spite of the license limitations (see Section 5.2.3, “License key considerations” on page 74).

NSA creates databases and reports based on its scanning. Depending on the number of scan targets and on the level of report detail, this can result in substantial storage capacity need.

We have looked at a few real-life samples, and found that for a scan that did just host discovery, there was about 4 KB of data per host in the database. A moderately sized scan on a range of active hosts resulted in less than 8 KB of data per host.

This results in our first rough guess; expect approximately 2 MB of data for every 256 addresses. If you scan a class B sized network and enforce reports even if there are no findings, you can expect a modestly sized database. However, care should be exercised when scanning a network of that size in

one run. The settings used with this should be first tested with a smaller network in case the output is more than expected. The amount of disk space depends on how long the data needs to be kept. However, this can be resolved by using storage archives.

The generated HTML report files can also get quite large.

5.2 First steps

In order to get a first impression of NSA and to obtain immediate results, duplicate the testing in this book. You can find more details on the required steps in Section 5.2.2, “Installation” on page 71.

1. Obtain the NSA package.
2. Make sure you have a license file. Note: Risk Manager customers will get their license file together with the NSA package.
3. Install NSA and the license file. Follow the instructions in the readme file, the User’s Guide, or in Section 5.2.2, “Installation” on page 71. Chose a directory name and define the symbolic link NSA uses to access its environment.
4. Change to the directory where the executable resides (`/usr/local/nsa-something/bin`).
5. Type `./nsa scan localhost.`
6. Wait for a short while and watch the report scroll by.

Figure 25 on page 70 shows the start and end portion of what you are likely to see. Version numbers will vary.

```
# cd /usr/local/nsa-1.2.1/bin
# ./nsa scan localhost
Network Services Audit (NSA)
Version 1.2.1; Apr 24 2000 07:15:12 [AIX 4.1.3.0 power]

Copyright (c) 1996-2000 International Business Machines Corp.
This product is proprietary material belonging to International
Business Machines Corp.  Disclosure to, or possession by,
unauthorized entities is prohibited.

Password:
Time until completion is less than 1 minute.
                                                    Network Services Audit Report
Network Services Audit Report

   Report Date: Tuesday, July 25, 2000 17:40

      o Name: localhost.austin.ibm.com (localhost)

        Operating System: UNIX IBM AIX 4.3
        Audit Date: `Tuesday, July 25, 2000 17:40'
        Auditor: `root@venus.itsc.austin.ibm.com'

       Security Audit Summary

         o Configuration Settings - 1
         o Dangerous Servers and Services - 4
         o Information Leaks - 5

... several pages of report data omitted for this screenshot

       o Active Users
            o rusers/32804 shows the following users logged on:

root <-
root <- 192.168.1.2
root <- :0.0


         o Server Banners
            o [23/TCP] telnet server banner -

telnet (venus.itsc.austin.ibm.com)

AIX Version 4
(C) Copyrights by IBM and by others 1982, 1996.
login:


            o [21/TCP] FTP server banner -

220 venus.itsc.austin.ibm.com FTP server (Version 4.1 Fri Nov 19 18:18:48 CST 19
99) ready.
```

Figure 25. Running a quick initial scan

5.2.1 How to obtain NSA

Registered users of Risk Manager will get an access password for the Tivoli Customer Support - Secure Downloads Web page. This can be found at `http://www.tivoli.com` by selecting **Support** from the horizontal menu at the top. On the next page select **Downloads** from the vertical menu on the left, and finally choose **Software downloads (for registered users)**. This brings you to a list of links to individual download pages. Currently (at the time of writing this book) the URL to bypass these steps is `http://www.tivoli.com/support/secure_download_bridge.html`

One of the links on this page will point to the NSA download.

5.2.2 Installation

NSA is distributed as a tar file, for example `nsa-1.3.tar`. This file maintains the directory structure, quite similar to the ZIP format that is popular on PC systems. Once you have placed that file on the system where you wish to install NSA, you must do the following as root:

1. Type `cd /usr/local.` (Create subdirectory `local` if it does not exist.)
2. Type `tar -xvf nsa-1.3.tar`. The filename may differ, and may include a path depending on where you stored the package file. The `-v` option is not essential; it makes tar display some messages (verbose mode).
3. `Type ln -s /usr/local/nsa-1.3/etc /etc/nsa.` This creates a symbolic link that points to the same physical location. NSA uses `/etc/nsa` to access its environment (unless this is overridden by a command line option). The following files *must* be found in that environment, so do not remove them:
   - scannerdefs
   - scannerdata
   - scannerrules
   - messages.cat (for reporting only)
   - at least one report template, typically under /etc/nsa/reports (for reporting only)

   There are other files that are used by NSA, but it will be able to function without them. At a later time you can always modify most of this environment if there is need.
4. Make sure that the license file `license.nsa` can be found by NSA. With the Risk Manager distribution it should be in place already. A search is carried out in this order:

   a. `$HOME/.license.nsa` (hidden file in user's home directory)

b. `$HOME/license.nsa` (visible file in user's home directory)

c. `/etc/nsa/license.nsa` (file in NSA's environment)

The above search can be avoided by using the configuration file directive `license /path/to/license`, which in turn can be overridden by the command line parameter `--license=filename`. This is a good first example how command line, control files, and default assumptions play together.

This allows for multiple users with different licenses sharing the same installation of NSA.

5. NSA can be used from this point on. However, for convenience you may want to add the directory where the executable resides to the path.

5.2.2.1 Special considerations for non-root users

NSA can be run with non-root user authorization. However, it needs to have root access in order to create a raw socket for ICMP tests, as well as binding sockets to lower port numbers. It will complain gracefully if it can't do what it needs to do, and continue to do what it can do (see Figure 26). This will happen shortly after the program has started to run. You can always decide to cancel the execution at that point and restart with root authorization.

A good practice would be to log on as non-root user, obtain root privileges using the `su` command, and then execute NSA. However, if none of the functions that require root are to be used, there is no need to invoke `su`.

```
ICMP: unable to create RAW socket: Permission denied
Can't perform tests for ICMP backdoors without ICMP socket.
rlogin: unable to bind socket in range 513-1023: Permission denied
rshell: unable to bind socket in range 513-1023: Permission denied
sunrpc: unable to bind socket in range 513-1023: Permission denied
bootp: unable to bind socket to port 68: Permission denied
netbios-ns: unable to bind socket to port 137: Permission denied
AIX-SRC: unable to bind socket in range 513-1023: Permission denied
```

Figure 26. NSA messages when invoked under non-root id

5.2.2.2 Caveats for non-UNIX users

For users who have worked with some kind of UNIX based system before, NSA does not pose any challenges. They can safely skip this paragraph.

Even those who have no previous UNIX experience should not find it difficult, either. No in-depth knowledge of UNIX is required to be able to use NSA. However, some considerations need to be taken into account, such as the subtle differences between a DOS command line prompt and a UNIX command shell:

- When you have done the installation you have already mastered the parameters generally required to unpack a tarred file. Normally by convention, but not necessarily, such a file ends with `.tar` . Be aware that the concept of file extensions does not exist under UNIX, so this is just as much a part of the file name as anything else. Usually you use the options `-xvf` to unpack such a file.
- If you have to investigate parameters and no reference manual is available, try `man <command>`. This will invoke the online reference base *man page* for the command in question if it is installed on your system.
- Unless you have added the path where the NSA executable resides to the `$PATH` environment variable (the way to do this may vary depending on the platform) you will most likely not be able to invoke NSA by simply typing `nsa` from its directory. You will have to type `./nsa` instead. Most UNIX systems do not have the current directory in the active path, and it does not get searched by default. This is for security reasons and should not be changed.

  Do not get irritated when you get the error message

  ```
  ksh: nsa: 0403-006 Execute permission denied.
  ```

  or something similar instead of

  ```
  ksh: nsa:  not found.
  ```

  that you expected in such a case. It still refers to the situation described here (The sample was obtained under AIX).
- If you create a non-root user to run NSA, you may have to use the `chmod` command to provide access rights to that user.
- Unlike DOS, under UNIX the command line gets processed by the shell first before the operating system interprets the command. This may result in modifications of the command line causing unexpected results. To avoid such effects, protect relevant parts of the command line from the shell using single quotes. Here is an example:

  ```
  nsa report -d site1.nsa '*.example.com'
  ```

  The asterisk has a special meaning for the shell as well as for NSA. By protecting it from the shell, it gets passed on to NSA unchanged (without the quotes).

- You will need a text editor to view and modify the control files. Most UNIX people use `vi`, because it is available under every UNIX system and gets installed by default. It is powerful but not intuitive. If you want to use it you will have to learn a bit about its fundamentals, and you must teach yourself at least a few basic commands and keystrokes. Obviously there are other editors, but none of them is as ubiquitous as vi. However, on any graphical desktop, be it CDE, KDE, GNOME, or other, there will be at least one alternative available.
- In several places of its configuration NSA allows regular expressions to be used. If you want to customize any of those (for example scanner data, report templates) you may have to learn about regular expressions in UNIX.
- In both policies and reports you will find conditional statements. Conditions test some kind of expression that can either be true or false. Multiple expressions can be combined into a compound statement using `&&` (logical AND) or `||` (logical OR). If the condition is a compound statement, the expressions must be enclosed with parentheses.
- If you want to run NSA on an automatic schedule, you should familiarize yourself with the `crontab` concept of timer driven automated program launch.

5.2.3 License key considerations

Registered Risk Manager customers will get download access to an NSA package that contains a license key that allows scanning of the local subnet in which NSA itself resides. A possible utilization of this could be a Web farm running on the same subnet as NSA - it can be checked periodically to assess vulnerabilities of the Web servers.

A full-blown corporate network that spans many subnets, however, cannot be covered by one NSA system with this license. For advanced utilization of NSA's capabilities beyond this scope, users may contact IBM Global Services. This does not imply that related service offerings are available in all countries.

5.2.4 A quick scan's output

In Section 5.2, "First steps" on page 69 we have initiated a brief scan of the local host. We did not provide any parameters regarding the output, so we received it on `stdout`, the unix default output device, which in this case was the console screen. We will now take a closer look at that output.

Console output, text file

In order to preserve the report we ran the same scan again, this time using the parameter `--outfile=firstscan.txt`, which has an additional effect as compared to redirecting stdout to a file; NSA will write the reports directly to disk, which saves the memory needed to build them.

A text style report does not allow much formatting; it is basically an unordered list with several levels of indentation. It actually consists of several reports with different levels of detail. At the top it provides information about the report itself and about the scan run that generated the data. This includes the name of the system where NSA was executed, the user id that was used (this one was just a test, so we have an excuse for using root), the operating system, and the date of execution. All this may sound trivial, but NSA is designed to be heavily used in a large scale environment, and therefore needs to keep track of those details, even more so when automatic processing of the generated data is done.

The actual reports are structured as follows below. Each report has at least one more level of detail than the previous one. The third and last report goes down to a granularity of five levels of indentation and shows every single finding.

The structure of the report corresponds to the categories of finding identifiers in the findings tree (compare Section 5.6.3.2, “Finding identifiers” on page 117). Below we provide the categories that appeared on our sample run. We matched them against the category names in the findings tree. You can ignore these if you just want to get an overview at this time, you may then want to revisit here when you read about finding identifiers.

The top two report types only address those findings that are considered risks, whereas the third category contains all findings, regardless of the level of threat they represent:

- Security audit summary
- Security audit breakdown
- Security audit findings

The latter (security audit findings) contains all the details. It breaks down further into the following:

- Configuration settings (Findings Conf325)
  - Access control configuration
- Dangerous servers and services (Findings Danger325)

  - Dangerous network servers (here: rshell, rlogin, several Sun RPC servers)
- Information leaks (Findings InfoLeak325)
  - Information about user accounts
  - Information about system resources (here: SNMP, portmapper, and others)
- Active network servers (complete list of servers found - Findings SysServers325)
- Available network services (Findings SysServices325)
  - User login services (complete list)

From here on the remaining report parts fall under the finding category Data325. We did not have any findings in the last category Vuln325 (vulnerabilities).

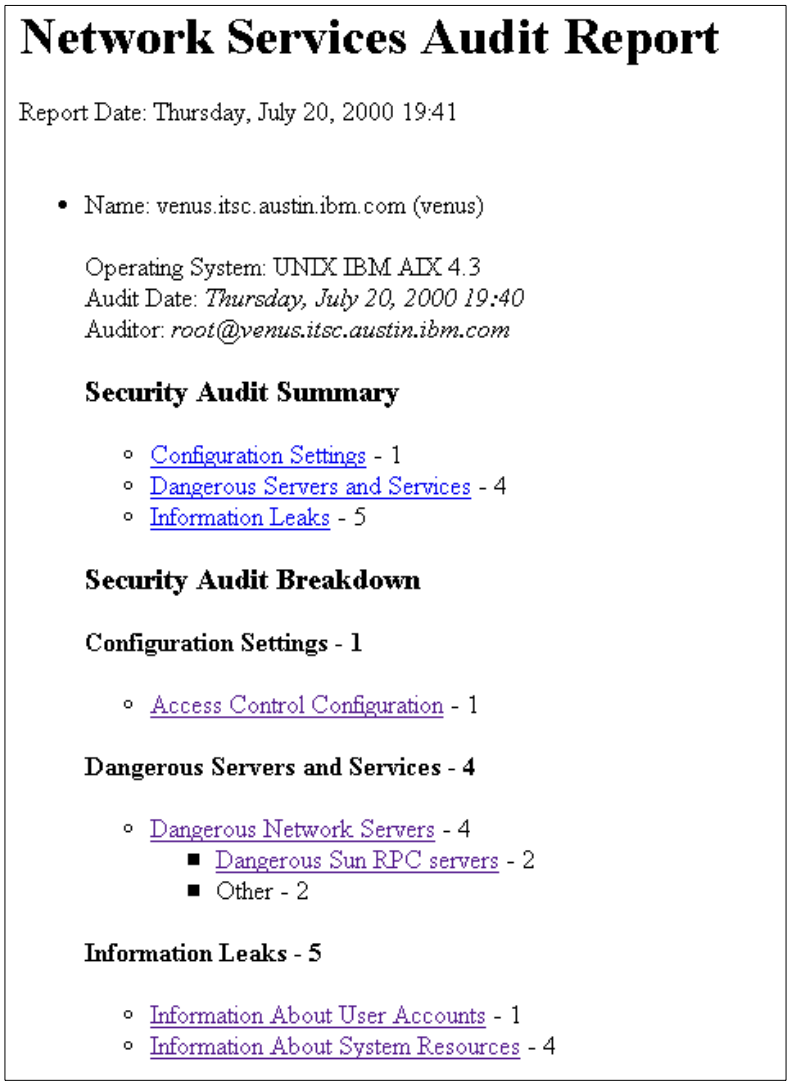
- Operating system (findings sub category SystemOSName)
- Server version strings (findings sub category ServerVersion)
- SNMP findings (here: community name public - findings sub category SNMPCommunity)
- Network transport information (here: responds to ping - findings sub category NetTransport)
- Port scan information (findings sub category PortScan)
  - TCP port scan data (ports scanned, ports visible, ports active, unidentified servers, servers that terminated immediately)
  - UDP portscan data (ports scanned, ports visible, ports active, unidentified servers, ports that did not respond)
  - Sun RPC registrations
- SNMP variables retrieved (findings sub category SNMPVariable)
- Active users (information obtained by utilizing one of the leaks found - findings sub category ActiveUsers)
- Server banners (findings sub category ServerBanner)
  - Telnet
  - FTP

Note that this is just an example based on the findings which happened to occur with this target host.

HTML output file

We have just seen that a default report can provide large amounts of data, but it is difficult to use. To find the most important pieces quickly, use the HTML report format. It provides better readability via a browser by using HTML markup for the report structure. It also eases navigation down to the critical findings by employing hypertext links from one report type to the next, more detailed level. Below we have provided a screen capture to demonstrate both of these advantages (Figure 27 on page 78). Note that this is only the first part of the report. It corresponds to the text report in Figure 25 on page 70.

A strong feature of HTML is its capability to embed external references via links. NSA comes with a number of index files (in subdirectory `referdata`); more can be added by the user.



Network Services Audit Report

Report Date: Thursday, July 20, 2000 19:41

- Name: venus.itsc.austin.ibm.com (venus)

  Operating System: UNIX IBM AIX 4.3
  Audit Date: *Thursday, July 20, 2000 19:40*
  Auditor: *root@venus.itsc.austin.ibm.com*

  Security Audit Summary

  - Configuration Settings - 1
  - Dangerous Servers and Services - 4
  - Information Leaks - 5

  Security Audit Breakdown

  Configuration Settings - 1

  - Access Control Configuration - 1

  Dangerous Servers and Services - 4

  - Dangerous Network Servers - 4
    - Dangerous Sun RPC servers - 2
    - Other - 2

  Information Leaks - 5

  - Information About User Accounts - 1
  - Information About System Resources - 4

Figure 27. HTML formatted NSA scan report

5.3 Working with databases

Now we will take a closer look at the NSA database, also referred to as findings database or on-disk database. We will discuss its benefits and its structure from a user's perspective.

5.3.1 Why databases?

The term on-disk database implies that there is another kind of database, the anonymous database. If you run a scan without specifying an on-disk database, NSA builds a database anyway, but it is invisible to the user and is deleted when the scan has finished.

Note

If you work with an anonymous database, this database is written to the current directory. You must have write access to that directory; otherwise NSA will fail.

Figure 28 illustrates the pivotal role which the database plays in the operation of NSA. By creating it on disk we achieve a number of benefits. It is therefore good practice to work with one or several databases.

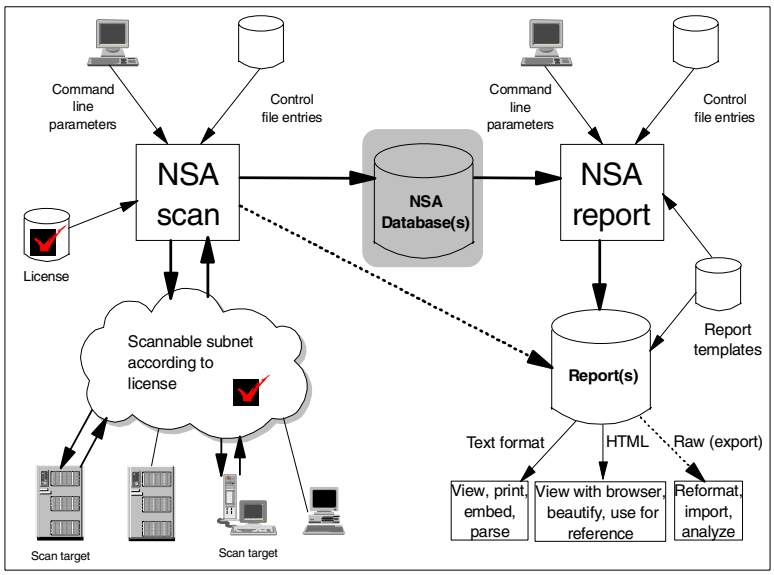


Figure 28. NSA database - the central entity

The benefits include the following:

- All findings are kept in a compact format in one place.
- Any kind of additional reports beyond the routine processing can be obtained any time later if a special need arises.
- Findings can be stored across multiple databases and thus can be grouped in a meaningful way. Examples could be:

- One database holds the weekly scans, one holds the monthly scans, and one or multiple databases hold any special scans.
- Scans can be arranged by subnets or subdomains.

- Delta reports can be generated that only show changes compared to previous scans. These previous scans have to be available in a database.
- If the need arises, multiple databases can be merged back to one again.
- While it performs a scan NSA can create copies of the database on the fly. This is called *mirroring*. Mirrored databases can serve as backups or as a source for intermediate reports while the scan is still running. This can be important because a scan may run for a long period of time (days or even weeks) depending on the selected options, and an interim report would be useful.

5.3.2 Structure of the database

A database consists of one or multiple scan groups. A scan group consists of one or multiple host scans. It is the result of a single run of `nsa scan`. A host scan holds the findings for a single host obtained during one run. There is one host scan entry in the scan group for each host that was part of the corresponding scan run.

In the example of Figure 29 a single scan adds a new scan group to two databases (weeklydb and newdb) and creates a new database (specialdb). Note that target host specifications following a database parameter have their findings stored into that database until another database is specified. This is what the NSA User's Guide means when it talks about positional parameters.

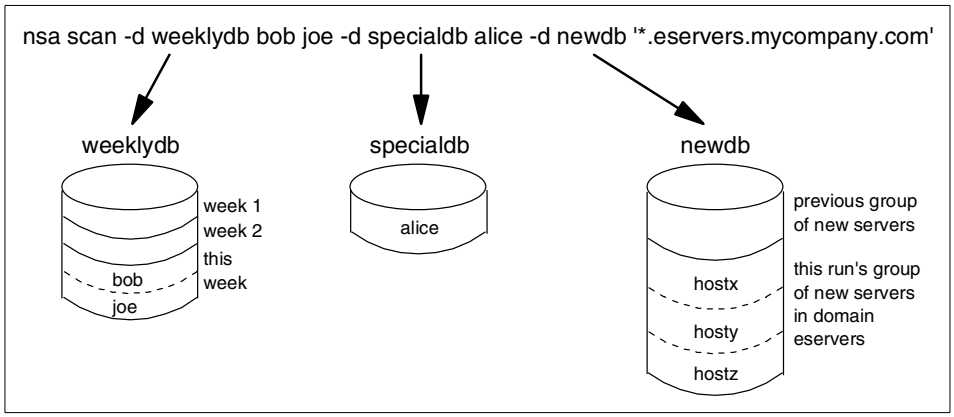


Figure 29. Structure of NSA databases

5.3.3 Utilizing databases at report time

All scans stored in a database can be accessed for reporting. If you do that with a single command, you will get one report which combines the selected scans in its high level sections, but has an individual detail report for each selected scan. If you want completely separate reports, use the option `--separate`.

The sample in Figure 30 takes the databases produced above and generates reports out of these. We are assuming that bob was scanned in the previous week as well. We decided to get all reports from week 1, bob's previous report and joe's latest report. In specialdb there is only one scan group with a single scan, so we do not have to specify anything to get this. In newdb we want everything that is in there, no matter if there are duplicate host names or not. If we had only specified '*' we would only have got the most recent scan for each host.

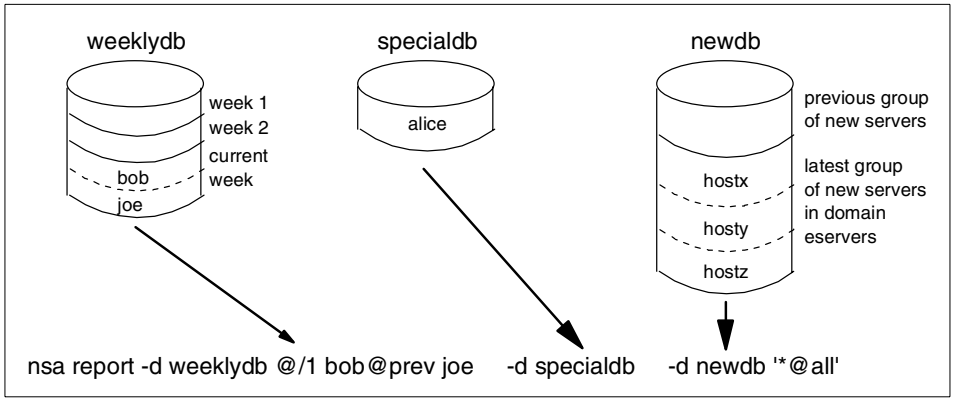


Figure 30. Accessing specific scans in a database

Another way to select scans is by using the scan date. A wide range of date formats is supported. For more details about accessing specific scans refer to a section called "Selecting Entries from a Findings Database" in the *NSA User's Guide*.

5.4 A full scan

Following the User's Guide we did the same scan as in Section 5.2, "First steps" on page 69, just adding `--scantype=complete` on the command line. The first noticeable difference was the duration of the scan. Instead of one minute it took three minutes to complete. The size of the report was not an indication; in fact it could even have been smaller due to the fact that it is easier to

summarize a range of port numbers on one line rather than having to list them individually.

The report (Figure 31) shows that the full range of ports has been scanned. So this is a much more complete picture of the target host, you can feel much safer about this host now. Combining this part and the report on active services, you now get a full view of this host's setup:

```
o Port Scan Information
          o TCP Port Scan Data
             o The following TCP ports were scanned:
               1-65535
             o The following TCP ports were visible:
               1-33676, 33678, 33680-34588, 34590-65535
             o The following TCP ports were active:
               7, 9, 13, 19, 21, 23, 25, 37, 111, 199, 512-514, 543-544,
                 974-975, 2401, 6000, 6112, 32768-32769, 32771-32772,
                 33676, 63488
             o The servers on these TCP ports could not be identified:
               199, 543-544, 6112, 32768, 32771, 63488
             o The servers on these TCP ports terminated immediately:
               None

          o UDP Port Scan Data
             o The following UDP ports were scanned:
               1-65535
             o The following UDP ports were visible:
               1-8, 10-136, 138-513, 515-32807, 32809-49576, 49578-65535
             o The following UDP ports were active:
               7, 13, 19, 37, 111, 161, 177, 518, 974-975, 32798,
                 32803-32807, 32834
             o The following UDP ports did not respond:
               9, 137, 514, 32808, 49577

          o The servers on these UDP ports could not be identified:
               37
```

Figure 31. Output from a complete port scan

5.4.1 Hotkeys

As we have seen in this example it can take a while for a scan to complete. This is a good moment to introduce the hotkeys that NSA accepts during execution of a scan. These allow the user to obtain information about the current state of the scan, as well as to abort the scan.

Keyboard controls are only active if NSA's input and output are attached to a TTY (keyboard and display). Thus, if you use shell redirection to redirect the output, the keyboard controls are inactive. However, the `--monitorport` option can be used to allow monitoring via a separate TCP connection.

The following keyboard controls are available:

Table 7. NSA keyboard controls (hotkeys)

Key combination	Invoked function
Ctrl-T	Estimated remaining time
Ctrl-X	Show status
Ctrl-P	Show tasks
Ctrl-C	Cancel scan (gracefully), no report
Ctrl-K	Shutdown scan (cancel remaining scans, terminate normally)
Ctrl-W	Write database copies

5.5 Configuring NSA

NSA can be configured to a great extent. This is achieved by using text files that reside in the NSA environment directory, which is normally `/etc/nsa` (This is why we discussed the topic of having a text editor earlier in this chapter). These files interrelate with command line options; they act as a default that can be overridden from the command line. In such cases there is a command line option and a corresponding *directive* in a control file. Even the names for most of the files are not a given and can be changed, but it is generally a good idea to leave them at their default unless there is a specific reason to do so.

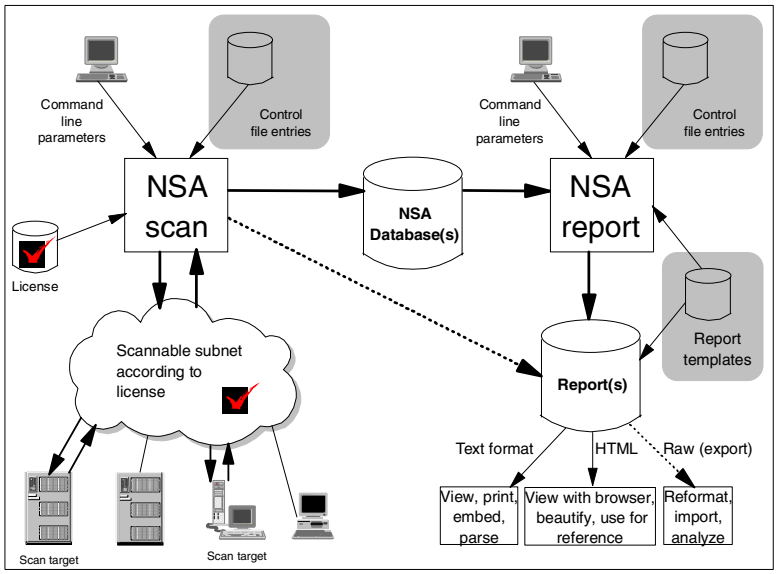


Figure 32. NSA control files

We will briefly describe the files here and then take a closer look at the most important ones. For a full description of all control files, refer to the *NSA User's Guide*.

- **Host files** are not configuration files in the true sense of the word. They provide lists of hosts on which NSA will operate both in scan and report mode, and they allow pre-setting certain options for those hosts.
- **Scanner definitions** are the actual definitions of scan activities that NSA is instructed to carry out. The scan activities include TCP and UDP port scans, built-in vulnerability tests by name, and randomized port scanning. Definitions can be nested and given meaningful names by which they are invoked. Normally they reside in the file `scannerdefs`.
- **Scanner directives** correspond to the built-in tests that NSA can carry out. They provide additional data for such tests (where applicable). They reside in the `scannerdata` file and are used during scanning only, not during reporting. Let's assume that a test for weak passwords is to be run, determined by a scan type defined in `scannerdefs`. Here is where you specify which user ids and which passwords you want NSA to try.
- **Policy definitions** allow to define what servers and services are allowed, as well as how severe a violation is. Policies are checked when the policy option is specified. They reside in `/etc/nsa/policies`. The default is "no policy".

- **Policy score violation mapping files:** During report generation, policy violations are displayed with a violation score, which is a numeric value. NSA allows defining a text string for a score value. If a mapping exists for a violation score, the text will be displayed instead of the numeric value. By default, no mapping file is used.
- **Report templates** are used at report time. There is a number of them being shipped with NSA, examples of which generate a standard report, a policy violation report, or a report that contains vulnerabilities. The templates use *built-in macros* to dynamically generate and insert data from the findings into the report. They reside in `/etc/nsa/reports`.
- **Reference information:** A report can generate references to other sources of information. These could be local text files or URLs. A built-in object in a report would point to a reference entry that gets embedded into the report as a text block or as a URL, depending on its type.
- **Configuration file:** This file contains configuration information that is similar to the command line options. The specifications therein override any default values, and are in turn overridden by corresponding command line options. NSA reads two configuration files, `$HOME/.nsarc` and `/etc/nsa/config`. The files are not read until needed.
- **Port to server mapping** files provide information to NSA about what server to expect on a particular port. NSA has a built-in set of mappings. This set can be extended on a per-host basis using mapping files.
- The ***t*ransmit speed names** file allows one to use symbolic names when specifying the port scanning rates. Depending on the network media type and the capacity of a target host, NSA should limit the rate at which ports are scanned in order not to disrupt a productive environment. This is defined here. The default file name is `xmitnames`.
- **Service names:** The `services` file contains mappings of TCP and UDP names to port numbers. It is only used when resolving names specified in a scanner definition or on the command line.
- The **Sun RPC names** file is used to translate RPC names to program numbers and vice versa. It is also used as the source of RPC program numbers for guessing what RPC program is listening on a port. The default file name is `rpc`.
- The **server task timer file** controls how long various server tasks wait for data when communicating with remote servers. The default file name is `tasktimers`.
- The **scanner rules file** contains rules used by the NSA scanner while performing tests. The file is typically at `/etc/nsa/scannerrules`.

5.5.1 Specifying scan targets in host files

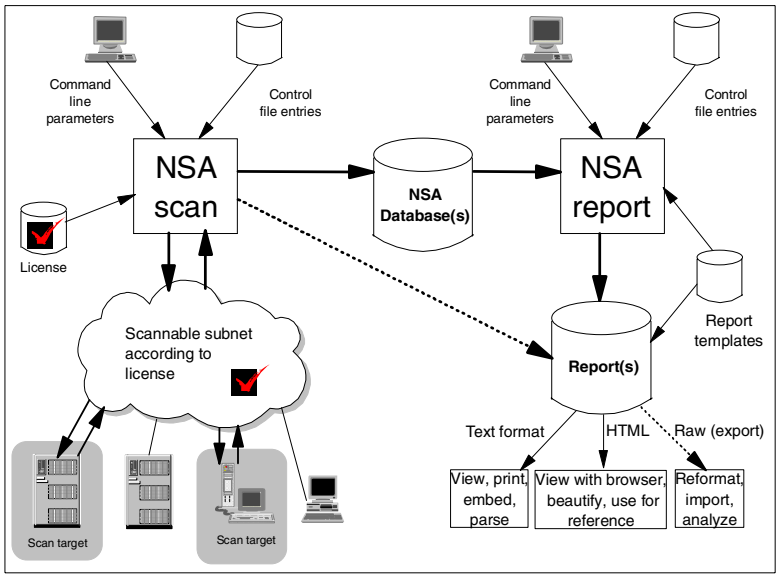


Figure 33. NSA scan targets

Host files are useful if you regularly scan the same set of hosts with the same scan parameters. Host files can be used in two different ways:

- Scan all of the hosts in the file
- Control the scan parameters for hosts specified on the command line

In this section we outline the concept; for specific details, please refer to the User's Guide.

To use the first method, simply specify the hosts file on the command line using a `--hostfile` parameter. Do not include any targets after it on the command line. All of the entries in the host file will be scanned.

The second use of the host file is to use it as a control for addresses that are specified on the command line. NSA will search for a matching entry and use the settings from that entry.

The format of the hosts file is

```
address [hosttag|* [ database|* [settings]]]
```

The address field is the only required field. It is all that is needed to perform the first method. Addresses can be specified as host names, a network address, or a range of network addresses.

The host tag is a unique tag for the host. Hosts in the findings database are indexed under this name as well as under the host name. Think of it as a more flexible way to refer to individual hosts as opposed to IP addresses or fully qualified names. Use “*” if this is not used.

The third field specifies the database in which to store the findings of the scanner, or from which to retrieve findings for this host. An "*" in this field acts as a place holder if no database is wanted.

The `settings` fields allows to specify certain parameters for the host entry. Supported settings include scan type and policy. For a complete list of parameters, refer to the User’s Guide.

The settings can either be coded here or in a separate file. In the latter case the filename must be provided here instead. Any parameter set here overrides the corresponding setting that is currently in effect. In turn, command line parameters that follow after this hostfile is invoked will override what is specified here. The following note gives an example.

Note

Most aspects of how NSA operates can be set in multiple places. There is a certain method how NSA determines which setting to use. It uses the concept of *positional parameters*. The following example explains how this works. Assume the typed command is this (on one line):

```
nsa scan --scantype=default --hostfile=myhosts myhost3
--scantype=complete
```

We assume that the hostfile `myhosts` contains the following lines.

```
myhost1 * * scantype=standard
myhost2 * * xmitrate=T1
myhost3 * * scantype=standard
```

`myhost1` will get scanned with type `standard` (hostfile overrides the command line value `default`), myhost2 gets scanned with type `default` (no hostfile setting, so the preceding command line value applies) and myhost3 gets scanned with type `complete` (the following command line parameter overrides the hostfile value).

5.5.2 Scanner definitions

The `scannerdefs` file defines scan types by giving them a name and specifying the actual scanning activity they represent. Scan type definitions can be nested for convenience and maintainability. With NSA 1.3 the definitions can be held in different files that can then be included from `scannerdefs` (this name can actually be overridden with the `--scannerdefs` command line option).

The format of the file is

```
define name
definition-record-type definition-data
end
```

The *name* indicates the name that should be used with the `--scantype` option. Note that the name "default" has a special meaning; it is used whenever no scan type is specified for a particular host. The User's Guide explains this in detail.

The *definition-record-type* must be one out of a set of recognized commands. They include:

- Definition of port ranges to be scanned - `tcpports, udpports, rpcsvcs`
- Controls of probabilistic scanning - `tcprand, udprand, coverage`
- Invocation of specific tests - `options`. (This will eventually be replaced by switches.) Examples for tests that can be enabled with this command are:
  - `backdoors` - Check aggressively for any backdoor servers that NSA knows about. The test will check all ports for them, not only the ones where they are normally found.
  - `icmp-servers` - Enables testing for any servers that communicate via ICMP messages. These are usually unauthorized servers on compromised machines.
- Enabling or disabling of *switches*. Switches are used to turn on individual vulnerability and exposure tests, from the scanner definition file as well as from the command line. Every test will eventually have a unique switch to control it, making the `options` command obsolete. By default, switches are off. The switch specification can contain wildcards, specified via an "*". Examples:

  ```
  -enable nsa-vuln.overflow.*
  disable nsa-vuln.overflow.http-3
  ```

  This enables all switches starting with “nsa-vuln.overflow” except for the one explicitly disabled.

```
-enable cve-1999-0751
```

This enables the switch `nsa-vuln.overflow.http-1` using its CVE alias.

- Embedding other definition files or including nested definitions from within the same file - `include`
- Set the value of a scannerdef variable. See User's Guide for details.
- Defines an `alias` - an alternate name for the scanner definition. This can be used to give a more easily remembered name to a scanner definition, or, as NSA uses it, to relate standardized names such as CVE identifiers to scanner definitions.

NSA is shipped with a complete set of scanner definitions that will get you started. However, this is a very powerful mechanism to enhance the functionality of NSA, and to adjust it to specific needs you will probably create your own definition as you gain experience.

5.5.3 Providing details for a built-in test in scanner directives

The `scannerdata` file contains information used by the NSA scanner while performing tests. The file is typically at `/etc/nsa/scannerdata` but can be specified via the `--scannerdata` command line option. Note that the data in the scannerdata file is only used during scanning, not during report generation. Therefore changes to the scannerdata file will not cause changes to a report generated from a scan performed prior to the change.

The format of the file is

```
tag data
```

The standard built-in tags are listed and explained in the following table. This is to give you an overview of the capabilities as well as the flexibility of NSA. More details about them can be found in the User's Guide.

Note that there can be arbitrary tags to be used by self-written scripts. The scripts will then be able to retrieve those tags from the `scannerdata`. We were unable to test this functionality because it was not ready at the time of writing.

Table 8. NSA scanner data tags

Tag name	What the tag specifies
backdoor-server	Determines unauthorized servers
bad-server-version	Indicates versions of a server known to contain vulnerabilities

Tag name	What the tag specifies
bad-service-version	Indicates versions of an underlying service known to contain vulnerabilities
good-server-version	Indicates versions of a server that are believed to be "safe" (contain no known vulnerabilities)
http-cgi	Known bad CGI program
http-cgi-dir-set	Set of directories to search for bad CGI programs.
http-cgi-version-check	CGI program that should have its version checked.
http-file	Files that should not be retrievable via HTTP
http-protected-url	Documents that are normally password protected, causing NSA to perform HTTP username password tests.
http-url-of-note	Documents that are to be checked for, but are not considered a security concern
http-userpass	Username and password combination to use when NSA attempts to access a protected document via HTTP
info-sys-server	Server should be flagged as a source of information about the system it is running on
info-sys-sunrpc-program	SunRPC program should be flagged as a source of information about the system it is running on
info-user-server	Server should be flagged as a source of information about user accounts
info-user-sunrpc-program	SunRPC program should be flagged as a source of information about user accounts
shared-file-of-note	Filenames that are to be checked to see if they exist on shared filesystems
smb-userpass	Username and password combination to test using the SMB file sharing service
smtp-alias	Aliases that should not be present
smtp-bad-cmd	Server commands that should not be present
smtp-bad-from	E-mail "from" values that are unacceptable
smtp-bad-rcpt	E-mail "recipient" values that are unacceptable

Tag name	What the tag specifies
smtp-relay-pair	Pair of templates used to generate the sender and recipient addresses when performing SMTP relaying testing
snmp-community	SNMP read community string to use as a guess
snmp-read-var	If the read community string is successfully guessed, then attempt to read the specified SNMP variable for inclusion in the report
sunrpc-pmap-no-fwd	Specify a SunRPC program that the SunRPC portmapper (rpcbind) CALLIT procedure should not forward calls to.
tftpfile	File to attempt to retrieve through a TFTP server
userpass	Username and password combination to test
user-login-server	Server should be flagged as providing access to user accounts
user-login-sunrpc-program	SunRPC program should be flagged as providing access to user accounts

An example that users are likely to use is the user name and password testing. Here is an example for the syntax:

```
userpass {
   User="root"
   Password="root"
   Class="unix"
}
userpass {
   User="guest"
   Password="guest"
}
```

The tags `http-userpass` and `smb-userpass` follow the same syntax. NSA uses these entries to attempt logins with the applicable servers. The `Class` field can be used to allow these entries to be invoked by class name rather than individually.

These tags get invoked when a vulnerability test is carried out that involves logon attempts. For an example see Section 5.6.2.3, “Fine grain control of vulnerability testing” on page 107.

5.5.4 Specifying site policies

Policy definition files allow a site to define what servers and services they allow, as well as how severe a violation is. Policies are checked when the `--policy` option is specified. If used, a report can be produced that details all violations against the site policy.

Rules are specified either as `allow` or `deny` statements. Note that the policy engine uses a default deny system, so you must wild card an `allow` at the end of the policy definition if you have only specified `deny` rules.

There are two types of policy definitions:

- The first type allows a site to define which TCP and UDP ports should be visible or active. *Visible* means that a port is reachable by the scanner. A reachable port is one that generates a response from the host. This can either be output from a server listening on that port, or the equivalent of "Connection Refused" for the protocol. *Active* on the other hand indicates that a port has live servers on it. Any port that responds with user data is considered active.

  The format of this policy rule is:

  ```
  allow|deny tcp|udp visible|active <port-list>
  ```

  Here is an example from the firewall policy that ships with NSA:

  ```
  allow tcp visible 25,53,80,443
  allow udp visible 53
  ```

  This allows the specified ports to be visible, but none of them should be active on a firewall system.

- The second type allows policy rules to be defined based on the findings of the scanner using *finding identifiers* (FIDs) and conditional statements. For more details about FIDs see Section 5.6.3.2, "Finding identifiers" on page 117. This type of rule (dubbed FID policy rule) allows policies to be based on almost any of the findings the scanner may record, as well as on the data fields within those findings. In addition, deny entries are scored with a severity level. The format of the FID rule is:

  ```
  allow|deny[/score] fid <fidname> [conditions]
  ```

  Score provides a numeric value indicating the severity of a violation against a deny rule. *Conditions* test the values of fields associated with the FID. The only operators are `==` for equality, `!=` for inequality, `~` for a matching regular expression and `!~` for a non-matching regular expression. Conditions can be combined into a compound statement using `&&` (logical AND) or `||` (logical OR). If the condition is a compound statement, the two expressions must be enclosed with parentheses.

Again we take a look at the firewall policy for an example:

```
allow fid SrvActTCP (ServerName == "SMTP") && (Port == 25)
allow fid SrvActTCP (ServerName == "HTTP") && (Port == 80)
allow fid SrvActUDP (ServerName == "DNS") && (Port == 53)
allow fid SrvActTCP (ServerName == "DNS") && (Port == 53)
# HTTPS
allow fid SrvActTCP (ServerName == "SSLv2/Unknown") && (Port == 443)
```

It says that a firewall should allow SMTP, HTTP, DNS, and HTTPS on their default ports.

5.5.5 Policy score violation mapping files

During report generation, policy violations are displayed with a violation score, which is a numeric value. NSA allows defining a text string mapping of the score value. If a mapping exists for a violation score, the text will be displayed instead of the numeric value.

By default, no mapping file is used. The mapping file can be specified on the command line with the `--levels` option, or in a configuration file with the `LevelNames` directive. See the two examples below. Note, however, that the tags can contain HTML markup, which enhances the options for an HTML report.

```
# This file assumes values from 0-100, with lower values being more severe.
#        start    end
   low      67     100
   medium   66      34
   high      0      33
```

Second example:

```
<font color=green>low</font>       67     100
<font color=orange>medium</font>   66      34
<font color=red>high</font>         0      33
```

5.5.6 Customizing reports with report templates

NSA is intended to show what's wrong with your network and how it operates. When you explain it to others, the better you can taylor the output to your target audience's needs, the better the outcome will be. NSA provides flexible *templates* to taylor its output.

5.5.6.1 Standard templates

NSA provides pre-defined report types. You can leave them as they are, modify them, or build your own from scratch. A template is invoked by

filename via the `--format` command line option, or by the equivalent directive in the configuration file.

NSA comes with the following standard templates:

brief	Only describes the scan run
standard	Default, we used it in Section 5.2.4, “A quick scan’s output” on page 74
summary	Contains counters only
delta	Reports only changes from last run
policy	Contains only policy violations
vulnerable	Contains only vulnerabilities
vulnlist	Same as vulnerable, less verbose
raw	Raw dump, can be used as input for other applications
index	Creates an index report

5.5.6.2 Customizing templates

The best way to get an understanding of how templates work is to look at a sample. We selected `/etc/nsa/reports/standard` as a starting point because this is the template that produced the report shown in Figure 27 on page 78. When looking at the report templates you should revisit that report and match it against the source that created it. All we need to know about the syntax at this point is this:

- Comments start with “@#”
- Conditional blocks generate text depending on conditional expressions. They function as an *if-then-else* construct. “@[“ represents the *if*, “@]” terminates the block (*endif*), the optional *else* portion is initiated by “@|”.
- Built-in macros start with “@.” and are followed by their parameters in parenthesis (if any).

The first part of the report (heading and report date) is created by what we see in Figure 34 on page 95 in the upper part.

We then skipped a few conditional sections that did not apply to our initial scan example. At the end of the file we see a block that generates the report per host. All it does is generate a list (`<ul>, </ul>`) and call another template file `host.detail` where the actual work is done.

```
<html>
<head>
<title>Network Services Audit Report</title>
</head>
<body>
@#
@# Allow user to specify alternate heading using variable NSA_HEADING
@#
@[ @.var(NSA_HEADING) != "";
   <h1>@.var(NSA_HEADING)</h1>
@|
   <h1>Network Services Audit Report</h1>
@]

<p>
Report Date: @.todaysdate()<br>
...
... (skipped part of file that do not apply to our example)
...
@# For each host, call host.detail
@#
<ul>
@.foreachhost(host.detail)
</ul
```

Figure 34. Report template generating a heading

Next we look at `host.detail` (Figure 35 on page 96) to see how the per-host report is generated. First the template produces an item in the list that we just created. Four lines of text are created using some of the built-in objects. At the end a conditional statement generates the sections for the Security Audit Summary and the Security Audit Breakdown if there is data in the findings that requires those sections.

The next level of detail is processed in yet another set of template files, `findings.summary` and `audit.breakdown`. All of these files are by default retrieved from `/etc/nsa/reports` and `/etc/nsa/reports/sections`. We stop our review of the example at this point.

```
@# Emit a header for this host.  Includes the Delta Against date/time
@# if there are any delta objects for this host.  This will eventually
@# be emitted if the user requested deltas at all.
@#
<li>
<a name="host!@.hostid">@#
Name: @.hostname @[ @.hostname != @.hosttag; (@.hosttag) @]@#
</a><br>
<p>
Operating System: @[ @.hostosname != ""; @.hostosname @| Unknown @]<br>
Audit Date: <i>@.hostauditdate(@.arg(1,))</i><br>
@[ @.deltaauditdate(@.arg(1,)) != "";@#
   Delta Against: <i>@.deltaauditdate(@.arg(1,))</i><br>
@]@#
Auditor: <i>@.hostauditedby</i><br>
...
... (skipped part of file)
...
@[ (@.fidtreecount(Conf325,) != 0) ||
   (@.fidtreecount(VulnMajor,) != 0) ||
   (@.fidtreecount(Danger325,) != 0) ||
   (@.fidtreecount(InfoLeak325,) != 0);@#
   <h3>Security Audit Summary</h3>
   @findings.summary
   <h3>Security Audit Breakdown</h3>
   @audit.breakdown
```

Figure 35. Report template generating a per-host header

There are many built-in macros that can be used to create a wide variety of reports. HTML adds to the richness of this feature. It is the recommended way to create reports for immediate review.

Note

A policy containing HTML markup (as all the provided standard templates do) can still be used to produce text reports. The HTML markup will serve as formatting information where applicable, and will be ignored elsewhere.

5.5.6.3 Reference data files

One of the built-in macros is `.ref`. It gets resolved to a source of reference information with the aid of index files (found in `/etc/nsa/referdata`). In HTML this generally means that a URL gets inserted into the report document pointing to a web site or to an FTP server with more information about a certain finding. A straightforward example is a quoted RFC. The resulting link would be derived from an entry in an index file resembling

```
rfc1954 ftp://ftp.isi.edu/in-notes/rfc1954.txt
```

Note that this is an FTP link, as links are not restricted to HTTP.

5.5.7 Setting default parameters

Throughout this chapter we have discussed various ways to configure NSA using a number of files. We generically referred to them as control files. However, there is one file which is specifically called the *NSA configuration file.* All of its entries (called directives) correspond to a command line option (see Section 5.6, “NSA command line” on page 97). In many cases both have the same name; if not they are similar. The value set in the configuration file sets the default, but can be overridden by the corresponding command line option. However, in case of positional command line options, the matter is more complicated. Positional command line options only apply to scan targets that follow after them. So the values in the config file may still apply to some other hosts. For a discussion of positional parameters, see also Section 5.5.1, “Specifying scan targets in host files” on page 86.

NSA by default looks for the configuration file at `$HOME/.nsarc` (this enables a user specific version of this file) and then at `/etc/nsa/config`. The command line option `--configfile`, however, takes precedence over these locations.

5.5.8 Other control files

The remaining control files, listed in Section 5.5, “Configuring NSA” on page 83, are less likely to be modified by the user. Therefore we refer the reader to the User’s Guide.

5.6 NSA command line

NSA is invoked from a UNIX command shell prompt. The first parameter must be a *subcommand*, either scan or report. The subcommand is followed by the command line options. They specify the operations that NSA is to perform in this particular run. Last but not least, NSA needs to be told about the target hosts that it is supposed to scan or about which it has to report findings.

The flexibility of NSA is greatly enhanced by the use of control files. The use of these, in turn, is controlled by a set of command line options. Ultimately there is a sophisticated way of interrelating command line options and control file parameters and directives. It is good practice to provide settings that are rarely changed in control files (or have a limited number of these control files and pick the right one for each run via command line), whereas information that frequently changes, such as scan target hosts for a quick scan, is best provided directly on the command line.

Figure 36 shows the influential factors of the utilization of an option. Think of policy as an example. If it is not specified anywhere, no policy will be used, as there is no default. For other options, like scantype, there is always a default.

There is a specific control file, in this case one of the policy files in the `policies` subdirectory. The policy is defined here and it is given a name. In the `config` file you could have a `Policy` directive invoking this policy file. If the `--policy` command line option was specified, it would override the directive and invoke a different policy file instead.

If our example is a positional command line option (which `--policy` is not), there could be a mix of values applying to different subsets of hosts.

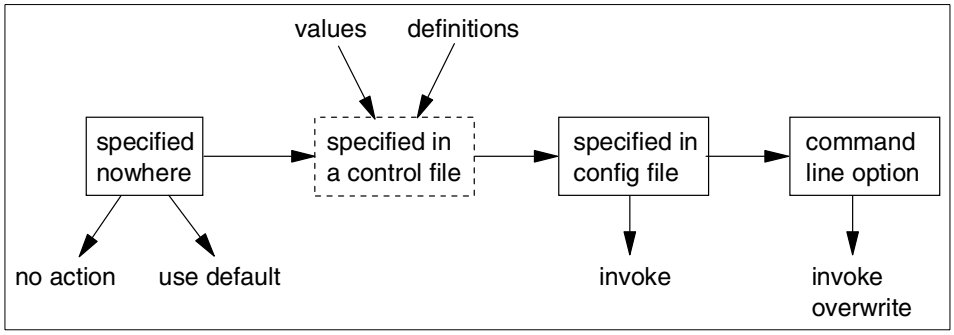


Figure 36. Relationship of control files and command line options

5.6.1 Overview

NSA accepts command line options in several formats. It is generally very flexible in the way options and parameters can be coded. However, we recommend that you use the preferred method until you have become very familiar with the product. For typing convenience there are short forms for some of the options.

The preferred format is the long form of the option. To specify an option in this manner, preface it with two dashes. For options that accept parameters, the parameter can be specified as part of the option using an '=' (equal) sign.

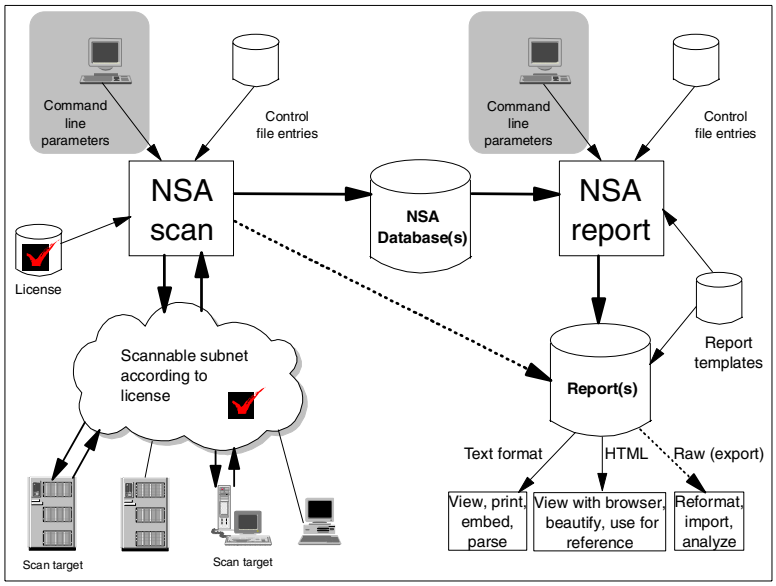


Figure 37. NSA command line

The options can be arranged in groups related to the type functionality. We have identified the following groups:

- Commonly available options
- Scanner control options
- Scan performance options
- Miscellaneous scanner options
- Report generation options
- Base configuration options
- Other options

Each group's features are detailed in the tables below. They also give information about short forms (where applicable), the corresponding control file (if any), and provide a flag for positional parameters. For a full description of the options, please refer to the User's Guide.

The following options (Table 9) are available no matter what mode of operation NSA is in:

Table 9. Commonly available options

Type	Name	Description	S	File	P
Set findings database	database	Specify on-disk db	-d	none (in-memory)	p
Set target hosts	hostfile	Read target hosts from file	-f	none	p

The following options (Table 10) are used to control what hosts and services are scanned, as well as what tests are performed.

Table 10. Scanner control options

Type	Name	Description	Short	File	P
Specify ports to be scanned	scantype	Scan definition by name	-	scannerdefs config	p
	tcpports	TCP ports to scan	-t +t --tcp	scannerdefs	p
	udpports	UDP ports to scan	-u +u --udp	scannerdefs	p
	rpcprogs	Sun RPC servers to scan	-r +r --rpc	scannerdefs	p
Invoke built-in functions	options	Special tests to perform	-o +o	scannerdefs	p
	enable	Turn testing switches on	-	scannerdefs config	p
	disable	Turn testing switches off	-	scannerdefs config	p

Type	Name	Description	Short	File	P
Fine tune scanning	forcelogin	Require all servers to login	-	config	-
	icmplog	Log all ICMP unreachable msg	-	config	-
	ipsrc	IP source address	-	config	-
	pingfirst	Ping before scanning	-	config	-
	tcpprescan	TCP ports for host discovery	-	scannerdefs	p
	tcpsrcport	TCP source port	-	-	-
	udpprescan	UDP ports for host discovery	-	scannerdefs	p
	udpsrcport	UDP source port	-	-	-
Limit scope of scanning	lite	Light weight scans	-	-	-
	nologin	Disable user login tests	-	config	-
	portmap	Port mapping	-	port mapping	-
	reset	Reset to empty scanner def	-	-	p
Probabilistic scanning	coverage	Percentage of ports	-	scannerdefs	p
	tcprand	Range of ports	-	scannerdefs	p
	udprand	Range of ports	-	scannerdefs	p

The following options (Table 11) can be used to slow down or possibly speed up the scan rate.

Table 11. Scan speed control options

Name	Description	Short	File	P
concurrent	Maximum concurrent hosts	-	config	-
icmpmax	Maximum outstanding ICMP Echo requests	-	config	-
icmpretry	ICMP retransmissions	-	config	-
icmptimeout	ICMP transmit timeout	-	config	-

Name	Description	Short	File	P
idletimeout	Terminating idle connections	-	config	-
maxfiles	Maximum simultaneous network connections	--maxfd	config	-
taskgap	Inter task connection time gaps	-	config	p
tasktimeout	Maximum server task run time	-	config	-
tcptimeout	TCP connect timeout	-	config	-
timelimit	Time limits for scanning	-	-	-
udpretry	UDP retransmissions	-	config	-
udptimeout	UDP transmit timeout	-	config	-
workingset	Working set size	-	config	-
xmitgroup	Per group maximum scan rate	-	config	p
xmithost	Per host maximum scan rate	-	config	p
xmitrate	Overall maximum scan rate	-i	config	-

The following (Table 12) are other miscellaneous options for controlling the behavior of NSA's scanner.

Table 12. Miscellaneous scanner options

Name	Description	Short	File	P
adaptive	Adaptive scan speeds	-	config	-
alwaysrecord	Record scan data for all targets	-	config	-
hostlimit	Limit total number of hosts scanned	-	config	-
norecord	Not updating database	-	-	-
noresolve	Avoiding DNS lookups	-n	config	-
quietrandom	Suppress reporting on randomly selected ports	-	config	-
random	Scanning ports in random order	-	config	-
randomscatter	Random scatter scan	-	config	-
scatter	Scatter scan	-	config	-
smtprelayfrom	SMTP Relay from address	-	config	-
smtprelayto	SMTP Relay recipient address	-	config	-

Name	Description	Short	File	P
timescale	Server read timeouts	-	server tasks timeout file	-

The following options (Table 13) are used to control the report generator. Note that there are no short forms available for any of these options.

Table 13. Report generation options

Name	Description	File	P
define	Defining user variables	config	-
dbcachesize	In memory database cache size	-	-
delta	Generating delta reports	-	-
filter	Filtering out findings	config	-
format	Specifying a report template	config	-
html	Raw HTML reports	config	-
levels	Specifying policy violation score to text mappings	config	-
mailto	E-mailing a report	config	-
outfile	Sending a report to a file	-	-
policy	Specifying a site policy	config	-
references	More detailed reports	config	-
report	Forcing report generation	-	-
separate	Separating reports	-	-
subject	Subject of e-mailed reports	config	-

The following options (Table 14) can be used to direct NSA to use alternate files. Note that if used, these options usually will need to be specified first on the command line, as NSA will resolve them to their default values (or values in configuration files) if any of them are needed during the parsing of the command line. So, in a way, they are all positional.

Note that these options represent a great deal of the flexibility of NSA, but they do not have to be used because there is an environment in place after installation that allows NSA to run.

Table 14. Base configuration options

Name	Description	Short	File	P
configfile	Location of configuration file	-c	-	-
directory	Directory of control files	-	config	p
license	Name of license file	-	config	-
licensewarn	License expiration warning time	-	config	-
localscandata	Local scanner control data file	-	config	-
localscandefs	Local scanner definition file	-	config	-
localscanrules	Local scanner rules file	-	config	-
messagecat	Messages catalog	-	config	-
policydir	Site policies directory	-	config	-
rpcnamefile	Sun RPC names file	-	config	-
referdata	Reference data index	-	config	-
referfiles	Reference data files search path	-	config	-
referenceserver	Reference server program	-	config	-
scannerdata	Scanner control data file	-	config	-
scannerdefs	Scanner definitions file	-	config	-
scannerrules	Scanner rules file	-	config	-
scripts	Script directories	-	config	-
sections	Report sections search path	-	config	-
services	TCP/UDP port names	-	config	-
tasktimers	Server task timers	-	config	-
templates	Report templates search path	-	config	-
xmitnames	Transmit speed names file	-	config	-

The following (Table 15) are various miscellaneous options that can help to make life easier for a user. There are no short forms.

Table 15. Miscellaneous options

Name	Description	File	P
mirror	Makes copies of active finding databases	config	-
monitorport	Remotely monitor NSA while running	config	-
password	Password on command line	-	-
showstats	Show run time statistics	config	-

5.6.2 Invoking a scan

The core function of NSA is scanning a set of computer systems, locating and recognizing the server tasks active on those systems, and performing tests for known security vulnerabilities on those servers. For that you type `nsa scan` followed by parameters that describe the scan to be carried out. The basic features are the following (given a pre-identified subnet to be scanned):

- Finding out what hosts there are in this subnet (*discovery*). Once you have discovered them (or if you already know them), you can specify the target hosts by name, network address, address ranges, or subnet specifications.
- Identify - for each target host - which ports are *visible* (packets appear to reach the target address, indicated by some kind of response, which could be the equivalent to “connection refused” as well as a positive response.
- Identify - among the visible ports - which ports are *active,* meaning they have live servers on them, responding with meaningful user data.
- Identify by name the active servers - *server type recognition*. This includes known backdoors that could have been installed accidentally or by intruders.
- Find *weaknesses* on the servers. This could be insecure versions with known vulnerabilities or weak login procedures.
- Simulate known *attacks* to test for vulnerabilities.

In addition to these basic functions, NSA has these advanced features:

- Customizable scanner definitions
- Fine grain control of vulnerability tests - specific built-in tests can be invoked by utilizing the `options` parameter and switches. Future

enhancements will add more switches, each of them representing a test for a specific weakness that can be enabled or disabled.

- User controllable scan rates - you can either run NSA as a high speed scanner, or reduce the level of disruptiveness that scanning imposes by definition onto your network.
- Network sampling - if a full scan is not feasible due to the network size, you can instruct NSA to select a certain number of targets out of a given range.
- Probabilistic scanning - a randomly chosen subset of a given port range will be used by NSA. Assuming this is done on a regular basis, there will be full coverage over time.
- Compliance with the CVE nomenclature.

We will now present in a few examples how NSA's features can be exploited.

5.6.2.1 Host discovery

The objective was to find all hosts in a given subnet, even those that do not respond to a ping because they underwent some hardening procedures. To achieve this goal we employed all available methods: ICMP, TCP, and UDP based scanning. We used the following command:

```
nsa scan --reset --pingfirst --tcpprescan=25,53,80 --udpprescan=53 10.1.2.208/25
```

Here is a closer look at the selected options:

- `--reset` creates an empty scanner definition and makes it the current scanner definition. We do not wish to perform any extra testing at this time that might be enabled by the default scan type.
- `--pingfirst` invokes the first of our three chosen scan methods, which is ICMP echo - the ping.
- `--tcpprescan=25,53,80` uses the ports that are generally utilized by SMTP, DNS, HTTP; services that could be expected on an otherwise hardened system.
- `--udpprescan=53` is the same as the previous scan for UDP, but only DNS is applicable.
- IP subnet `10.1.2.128/25` using NSA's *network designator* syntax. The number 25 indicates that the first 25 bits form the subnet part of the address, leaving 7 bits for the host part. This is equivalent to a subnet mask notation of `10.1.2.128/255.255.255.128`. We are selecting a subnet of 128 addresses.

The output was a standard report, showing all discovered hosts and scan results for the selected ports. Note that no other ports were scanned due to the `--reset` option.

5.6.2.2 Customizable scanner definitions

Assuming that the discovery method employed in the previous section is generally applicable, we could simplify future discoveries by defining a new scantype that could then in turn be invoked with a command line option.

We take the port selections and use them to define a new scantype in file `scannerdefs`

```
define discover
    tcpprescan 25,53,80
    udpprescan 53
end
```

Figure 38. A self-defined scantype

And we invoke them like this:

```
nsa scan --pingfirst --scantype=discover 10.1.2.208/25
```

We should see the same results as in the previous section. Note that we do not require `--reset` this time because a scantype is explicitly invoked.

5.6.2.3 Fine grain control of vulnerability testing

NSA uses the concept of *switches* to turn on individual built-in vulnerability and exposure tests. Every test will eventually have a unique switch to control it. By default, switches are off. They are turned on via the `enable` directive (or the corresponding `--enable` command line option), and explicitly turned off via the `disable` directive. The switch can contain wildcards, specified via an "*". The switches have names starting with `nsa` followed by a category string, so a group of them could be invoked via `--enable='nsa-acl.*'` for all access control tests related to weak passwords, or `--enable='nsa-vuln.ftp*'` for all FTP related vulnerabilities. Those switches that are related to CVE standardized vulnerabilities have an alias name which is the CVE name (See Appendix B, “Common Vulnerabilities and Exposures (CVE)” on page 199). So they could be invoked by `--enable='cve*'`. For a complete list of all switches, please refer to the User’s Guide.

In many cases there is a relationship between a switch and settings in a control file. Let’s look at an example:

The switch `nsa-acl.logins.other` checks for accounts with weak passwords in the class `other` as defined in the `scannerdata` file (compare Section 5.5.3, “Providing details for a built-in test in scanner directives” on page 89). If the class field of a `userpass` or `smb-userpass` entry is defined, then the switch to enable testing is `nsa-acl.logins.<classname>`. For example, if the class is `unix`, then the switch would be `nsa-acl.logins.unix`. The class for usernames collected via the output of servers such as "rusers" is set to "learned". The following command line has returned findings of weak user passwords:

```
# nsa scan --reset --scantype=default_settings --enable='nsa-acl.*' venus
```

Note

It is not sufficient to enable a switch; there must be an adequate amount of port scanning enabled, too. The latest enhancements have eased this for most of the switches (see the next section). However, the password example given here still required explicit scanning definitions at the point of writing.

5.6.2.4 Convenience scan types

As we have seen in the previous section, there is more to the use of switches than simply invoking them. In fact it may require a substantial amount of detailed knowledge to find the appropriate combination of scanner definitions to go with a switch. In many cases NSA is now taking care of that on behalf of the user. A switch is then accompanied by a built-in scanner definition of the same name as the switch itself. This scanner definition includes the required port scanning. Thus, to test for the IQUERY buffer overflow, one need only input:

```
nsa scan --scantype=nsa-vuln.overflow.dns-1 targethost
```

The switch comes with the following internal scan type definition, which in turn enables the switch when it gets invoked as shown here.

```
define nsa-vuln.overflow.dns-1
   enable nsa-vuln.overflow.dns-1
   tcpport 53
   udpport 53
end
```

Previously, one would have to have done something like:

```
nsa scan --reset -t 53 -u 53 --enable=nsa-vuln.overflow.dns-1 targethost
```

The method of invoking a switch by means of its associated scan type is best for most cases. Normal users would rarely need to invoke switches on the

command line. An example where you still would is if you had an HTTP server on, say port 8080 and you wanted to test for HTTP vulnerabilities. Because the switches use port 80 for HTTP, you'll need to build the test yourself:

```
nsa scan --reset -t 8080 --enable='nsa-vuln.http.*' target
```

The User's Guide has complete information about convenience scan types for each of the switches. These scan types include alias names for CVE names wherever applicable.

5.6.2.5 User control of scan speed

Let's first discuss how to speed up NSA to improve its performance. Then we will look at reasons why we sometimes want it to slow down again.

Speeding up NSA

NSA has several controls that affect the performance of scanning, which tends to be limited by the delay imposed by waiting for responses from targeted hosts. The more requests that NSA can have "open" at the same time, the less time it will spend waiting for responses to requests, and thus the faster it will run.

The most important option for controlling the amount of work NSA performs simultaneously is `--maxfd`, which controls the maximum number of network connections (socket file descriptors) that NSA can be using at one time. For example, to allow NSA to use 256 network connections (assuming the system allows this many), use the following:

```
nsa scan --maxfd=256 192.168.0.0/24
```

Increasing this option can drastically improve NSA's performance if it is spending significant time waiting for responses. Note, however, that increasing this too much can adversely impact the launch machine, as NSA will consume large amounts of system resources. Some systems are unable to recover once they become oversubscribed, requiring a system reboot. The default value is 64.

Spreading the load

Increasing the number of network connections that NSA can create simultaneously can provide a large boost in performance. However, if all of the connections are directed at a single target machine, the load generated on that machine may cause it to respond slowly, which will impact the performance of the scan (and possibly even the quality of the results).

Spreading the work load across multiple target platforms reduces this problem. By default, NSA scans only four hosts concurrently. Increasing this

will reduce NSA's impact on an individual host (assuming there are enough targeted hosts available to maintain the specified concurrency level). Use the option `--concurrent` to modify this number.

Slowing down NSA

Scanning at high speeds can saturate network links and network devices, or overwhelm the targeted hosts. This results in very poor results, as responses are not reliably received by NSA when this happens. There may also be an undesirable negative impact in case of a live production network.

NSA provides three controls for adjusting the speed at which port scanning occurs. These are:

- `--xmitrate` - This option controls the number of ports that can be scanned per second across the set of hosts that are being scanned concurrently. It is intended to prevent NSA from saturating any network links that are common to all of the targeted machines.
- `--xmithost` - This option controls the number of ports that can be scanned per second for an individual host. This option is positional and can be specified multiple times. It is intended to prevent NSA from overloading a targeted machine.
- `--xmitgroup` - This option controls the number of ports that can be scanned per second for a group of hosts that are being scanned concurrently. This option is positional and can be specified multiple times. It is intended to prevent NSA from overloading a network segment or network device across which, all traffic to a cluster of targeted machines must cross. For example, if there is a remote office that is connected via a slow serial link, this option can be used to cluster that group of machines together so that NSA will not saturate the slow link, while at the same time being able to scan the other systems at a faster rate.

These three options can be used together, the order of priority being `--xmitrate`, `--xmitgroup`, and then `--xmithost`. Also note that with adaptive scanning enabled, NSA will adjust an individual host's scan rate for UDP downward based on the responsiveness of the host.

Note

All the options mentioned here can (and should, in fact) be specified in the configuration file. By doing this you will set default values that can be overridden in special cases. This avoids the risk of forgetting critical settings by mistake, and keeps up the stability of your environment.

5.6.2.6 Network sampling

There are situations where complete coverage is not necessary. If a new vulnerability is discovered, and one needs to size the amount of work necessary to correct it, a full scan may take too long, especially on a large network. NSA allows addresses to be selected at random for scanning. Address sampling can be used to obtain a rough estimate of the exposure throughout the network. As an example, the following will select one thousand addresses at random from the 10.0.0.0/8 address space and scan them. The problem with this is that it is very likely that a large percentage of the addresses selected will not currently be in use. The host discovery features of NSA can be used to overcome this.

```
nsa scan --pingfirst --hostlimit=1000 --randomscatter 10.0.0.0/8
```

Instead of simply selecting one thousand addresses and scanning them, NSA will select addresses at random and attempt to send an ICMP Echo Request to each address selected. As ICMP Echo Reply messages are received, NSA will scan the "discovered" address. It will only do this for one thousand addresses. Thus, when host discovery is used, NSA will be scanning only active hosts when performing network sampling. Any of the other methods for performing host discovery (see Section 5.6.2.1, “Host discovery” on page 106) may be used.

5.6.2.7 Probabilistic port scanning

Often it is not feasible to attempt to perform complete scans of every TCP and UDP port when dealing with large numbers of hosts. This becomes even more of a problem when performing scans on a regular basis, such as weekly.

To assist with this problem, NSA can be configured to select a set of ports at random for scanning. When used as part of a regular scan schedule, this is referred to as probabilistic scanning. Over time, all TCP and UDP ports will be scanned. For example, to have NSA select five percent of the TCP ports in the range 1024-65535, the following can be used:

```
nsa scan --tcprand=1-65535 --coverage=5% 192.168.0.0/16
```

This will cause NSA to pick, at random, approximately 3,300 TCP ports to scan, in addition to any TCP ports specified in the default scantype. Each of the 64k addresses scanned will have a different set of ports selected. In this example, it would take 20 weekly scans to get a relatively complete picture.

To put it into perspective, attempting to scan all 65535 TCP ports on each address, at 2000 ports per second, would require at least twenty-four days to complete.

Scan definitions can be configured to include probabilistic port scanning as well:

```
define weeklyscan
    tcpports 1-1023
    udpports 1-1023
    tcprand  1023-65535
    udprand  1023-65535
    coverage 5%
end
```

Figure 39. Definition of a weekly probabilistic scan

5.6.2.8 Running NSA in combination with Risk Manager

We have considered two different scenarios of using NSA in combination with Risk Manager:

- Running NSA regularly - defined as an authorized scanner in Risk Manager
- Using NSA to test the Risk Manager setup

Defining NSA as an authorized scanner in Risk Manager

This is easy. It is described in Section 4.5.4, “Defining source hosts” on page 58 and Section 4.5.5, “Defining authorized scanners” on page 60. First you must define the host running NSA as a known source host, for example

```
source(2001,'venus','9.3.187.221')
```

This will obviously only work if you have a fixed IP address for the NSA machine. If you use NSA on a mobile system in different subnets, you must determine a fixed IP address for each subnet and create one entry for each subnet, all referring to the same token.

Next you define the authorized scanner itself by using the token you just created. You only need one statement here.

```
authorized_scan(2001).
```

This is all you need to do in order to avoid having NSA’s activities raise an alarm every time it runs.

Testing Risk Manager

You can use NSA to simulate attacks in order to test your Risk Manager environment, including the IDS systems and their setup. In particular you may want to run such a test when you have changed the configuration (for example a threshold value) or when you have installed new sensors. Obviously, in this case, you do not want NSA to be considered an authorized

scanner, so you either turn off that definition or you run NSA from a different IP address that is not defined in the Risk Manager configuration.

We picked an arbitrary example to simulate this scenario:

```
nsa scan --scantype=nsa-vuln.overflow.dns-1 10.1.2.128/25
```

We used ISS RealSecure as a sensor. The events were eventually forwarded to Risk Manager. Because we scanned a subnet of 60 targets, there was a three-digit number of events coming in. They all showed in the RiskMgr_All container. In addition, we noticed a change of the RiskMgr_Situations container (Figure 40).

Figure 40. NSA causes a Situation on the Risk Manager console

We clicked the icon to see what it was. It turned out to be a class A_Situation2 (Figure 41 on page 114). We looked at the event details and found that it was classed as a Service scan (Figure 42 on page 114) by the correlation engine. This example shows how NSA has been used to verify that the correlation mechanism of Risk Manager succeeded in correlating many different events into one consolidated alarm.

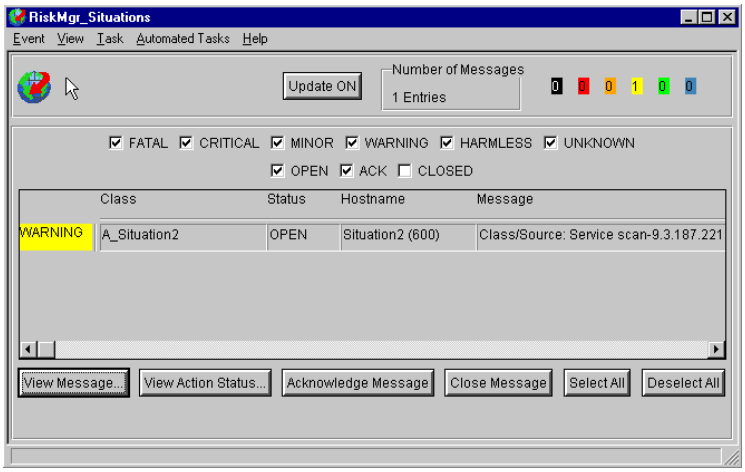


Figure 41. Situation2 caused by NSA

Figure 42. Details of the Situation2 event caused by NSA

5.6.3 Generating reports

A scan produces large amounts of information. In order to put this information to use, NSA has a reporting function.

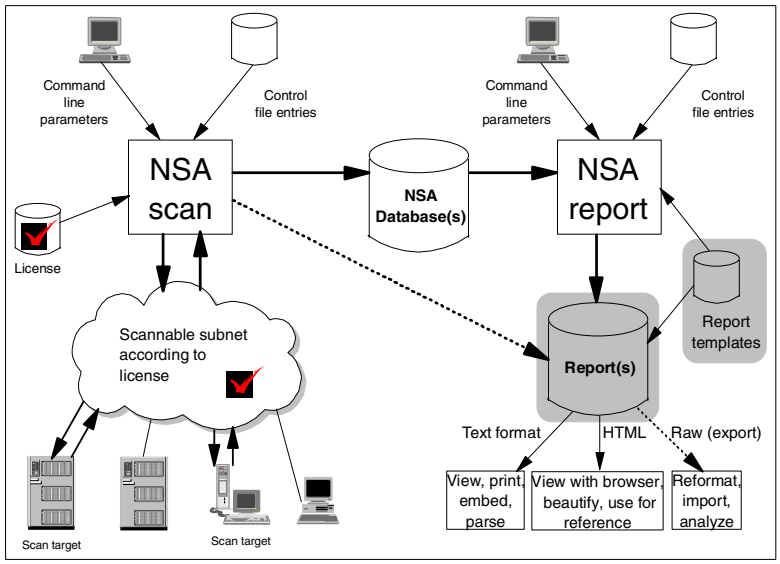


Figure 43. NSA reports

The reporting function is invoked by `nsa report`, followed by parameters that describe content and format of the desired report:

- Source of information - database(s) and selection of parts of their content. The basic unit of information is a scan, which is the result of one scan run related to a single target host. The scans created during one NSA scan run form a scan group. Your selection can be based on scan groups or individual hosts.
- Depth (level of detail) of report to be produced. There are preconfigured report templates. They can be modified and new ones can be created to suit any need.
- Format of report - the report can be either plain ASCII text, HTML, or RAW data format. The latter is intended to act as input for other applications that can exploit the data for further analysis.
- Invocation of additional, external sources of reference information.
- Policy definitions that the findings can be checked against for violations. A severity rating for the violations can be provided.

- Delta report generation - a comparison against a baseline report, preferably a full scan's output, that means only changes get reported.

5.6.3.1 Tailoring existing reports

In Section 5.6.2.1, "Host discovery" on page 106 we have carried out a host discovery with the intent only to learn all the IP addresses in the given subnet. So a single-line output with the IP address and name of each host that was found would have sufficed. The standard report gave us much more than that; it gave us a full report on each host.

We did not have the skills and time to create a new report template from scratch to achieve this goal. However, it turned out to be feasible to do a quick-and-dirty modification to the existing standard template. A few simple changes in two files created the desired format. Compare the results in Figure 44., "Host discovery condensed report" on page 117 to Figure 25 on page 70, which was created with the original `standard` template. Note the line

```
o Name: localhost.austin.ibm.com (localhost)
```

This is essentially all we want in our report with the IP address added. Firstly, here is what we did to `reports/standard`. We found the line where the host summary is invoked.

```
<h2><a name="hostdata.section">Host Audit Detail</a></h2>
<ul>
@.foreachhost(host.list)
</ul>
```

While `host.list` could have been modified, we instead replaced the line with

```
@.foreachhost(dis_host.list)
```

to avoid destroying the existing template. We then located the line where the detailed report by host is invoked. We commented it out because we did not want it.

```
@# For each host, call host.detail
@#
@# <ul>
@# @.foreachhost(host.detail)
@# </ul>
```

We saved the file as `dis_standard`, then edited `host.list` in the subdirectory `reports/sections` in order to adjust the single per-host output line. We found (one line in the original file)

```
<li><a href="#host!@.hostid">@.hostname - [@.fidtreecount(three25,)]
@.hostauditdate(@.arg(1,))</a>
```

and changed it to

```
<li>@.hostname (@.hosttag) - [@.fidtreecount(three25,)]
```

Finally we saved the file as `dis_host.list`, and invoked the discovery scan:

```
nsa scan --pingfirst --scantype=discover --format=dis_standard 10.1.2.208/25
```

(Compare Section 5.6.2.1, “Host discovery” on page 106 and Section 5.6.2.2, “Customizable scanner definitions” on page 107 to see how the command line has evolved.) This produced a report that contained a single line per host with host name and IP address in it.

```
...
   o venus.austin.ibm.com (9.3.187.221) - [4]
   o earth.austin.ibm.com (9.3.187.222) - [6]
   o jupiter.austin.ibm.com (9.3.187.223) - [6]
...
```

Figure 44. Host discovery condensed report

The number in square brackets is the number of findings. They are still there, but are suppressed in this report.

5.6.3.2 Finding identifiers

NSA records everything it finds using *finding identifiers*, or *FIDs*. FIDs are hierarchically arranged, but for most purposes, one does not need to know the structure of the tree. Many of the report generation templates however, are based on the structure of this tree. Therefore we chose to represent them in a table that is structured like a standard report (compare the following figure to Section 5.2.4, “A quick scan’s output” on page 74 to match the findings categories to the report sections). For a detailed description of each finding, please refer to the User’s Guide. If you are using the guide on-line you can search for the names given here.

Table 16. Finding Identifiers

Category			
Conf325 (Configuration settings)	ConfACL	User Accounts	NoPassword, WeakPassword, ByPassLogin, BadPassword, HTTPUserPass
		ACLXserver ACLFileShare OpenPrintShare MissingACL SNMPWeak OpenXDM OpenNNTP NISGuessable	
	ConfWritableDir		
	ConfWritable	WDirAnonFTP MKDirAnonFTP WFileAnonFTP	
	ConfSetting	SMTPEXPNVRFY SMTPRelay BINDIQUERY	
InfoLeak325 (Information leaks)	InfoUsers		
	InfoSystem		
	InfoDNS	DNSAXFR	
	InfoMisc	SunRPCPortMapperPort FTPAnonRealHome	

Category			
Vuln325 (Vulnerabilities)	VulnMajor	ObsoleteVersion	
		VulnFTP	FTPGuestFlag, FTPCHMOD, FTPDOTDOT, FTPCWDHOME, FTPPASV, FTPSITEEXEC
		VulnHTTP	HTTPCGI, HTTPFile, HTTPObsoleteCGI HTTPDOTDOT, HTTPIISMDAC
		VulnSMTP_dep2	
		VulnSMTP	SMTPBadAlias, SMTPBadCmd, SMTPBadAddress
		VulnTELNET	TelnetEnvVarValue
		SysBackDoors	RKDiscard, BackDoorServer
		VulnNetTrans	
		VulnMisc	KerbInfoLeak, BufferOverflow CFormatString, VulnVarious SunRPCPortMapCallFwd
		VulnNFS	NFSDotDot, NFSBigUID NFSmknod, NFSWeakFH NFSOpenMountd NFSMountdTrustsHostname
		VulnSMB	SMBDotDot
	VulnMinor	FingerDigit FingerNull FTPPortRedirect	
Danger325 (Dangerous servers and services)	DangerousService		
	DangerousServer	BadSunRPC	
	DangerousPort		

Category			
Data325	HostUnreachable SystemOSName SystemNISDomainName HostDown,NetBIOSName ServerVersion ServiceVersion TCPMUXServices NetBIOSScopeID NetBIOSGroup SMBShareSecurity TelnetEnvironment SNMPCommunity		
	NetTransport	NTLIP	TCPSeqPredictable, TCPSynFlood,IPForwarding IPShiftedFrag IPOptionsAllowed IPSrcRouteAllowed IPSiteAddressOK IPSmallFrag,ICMPInfo, ICMPEchoReply ICMPTimeStamp ICMPAddressMask, ICMPDomainName
	PortScan	TCPPortInfo	TCPPortsScanned TCPPortsVisible TCPPortsActive TCPPortsUnknown TCPPortsWrapped
		UDPPortInfo	UDPPortsScanned, UDPPortsVisible, UDPPortsActive, UDPPortsQuiet UDPPortsUnknown
		ICMP-Unreachable	
		PScanSunRPC	RegisteredTCPSunRPCSvc RegisteredUDPSunRPCSvc RegisteredSunRPCSvc
	DataFiles	HTTPDataFile, SharedFile	
	SNMPVariable FTPOddFilename ActiveUsers, ServerBanner		

Category			
SysServers325	ServerActive	SrvActTCP SrvActUDP SrvActTCPSunRPC SrvActUDPSunRPC SrvActTCPSunRPCUnknown SrvActUDPSunRPCUnknown SrvActICMP	
SysServices325	UnAuthFS	AnonFTP TFTPRead	
	FileShare PrintShare DeviceShare LoginServer		

Many of the FIDs have data associated with them. These can be referenced via a field name. The details of these FIDs are used most often when writing site policy definitions or report templates, or when dealing with raw report output data.

5.6.3.3 Providing input for other applications

These applications can produce output in plain ASCII text files or HTML files, but you would have to write your own program to extract the data because these formats are intended to be human readable rather than machine readable. The RAW format is much preferable for this reason.

RAW format

The RAW report format provided by the `raw` template provides a machine readable, structured way of presenting NSA's findings. It uses the Finding Identifiers (see Table 16 on page 118), and creates one line of output per finding. The format of each line is:

```
hosttag FID [field=value [field=value...]]
```

Note that some lines show an additional field between FID and the first field. It starts with `%KeyList=[` and is called a pseudo-field. At this stage it can be safely ignored.

```
venus AuditInformation HostName="venus.itsc.austin.ibm.com"
AuditDateSinceEpoch=965417099
venus AuditParameters
venus ReportData HostID=1
venus SystemOSName OSName="UNIX IBM AIX 4.3"
venus ServerVersion %KeyList=[Port,Protocol,ServerName] Port=21 Protocol="TCP"
ServerName="FTP"
venus SNMPCommunity %KeyList=[Port,ReadWrite] Port=161 ReadWrite="read"
Community="public"
venus ICMPEchoReply

venus TCPPortsScanned PortList=[21-23, 25, 31, 53, 58, 80, 98, 109-111, 121, 139, 143,
421,
venus TCPPortsVisible PortList=[21-23, 25, 31, 53, 58, 80, 98, 109-111, 121, 139, 143,
421,
venus TCPPortsActive PortList=[21, 23, 25, 111, 512-514, 974-975, 6000, 32769]
venus TCPPortsUnknown PortList=[]
venus TCPPortsWrapped PortList=[]
venus UDPPortsScanned PortList=[53, 69, 111, 137, 161, 177, 974-975, 1349, 2140, 6001,
10067
venus UDPPortsVisible PortList=[53, 69, 111, 161, 177, 974-975, 1349, 2140, 6001,
10067, 10666,
venus UDPPortsActive PortList=[111, 161, 177, 974-975, 32803-32807, 32834]
venus UDPPortsQuiet PortList=[137, 32808]
venus UDPPortsUnknown PortList=[]
```

Figure 45. RAW report sample

5.6.4 Other NSA subcommands

There are two more subcommands available with NSA that deal with databases. They are:

- `index`
- `merge`

5.6.4.1 List findings in a database

The `index` subcommand is used to obtain information about scans that are recorded in an NSA findings database. Currently, the only functionality available is to obtain a list of scan groups that have been recorded in the findings database. There is more flexibility built into this feature for future enhancements. A command line like this

```
# nsa index scans -d site1.nsa
```

would produce a result like the one shown in Figure 46 on page 123. Note that all scan groups have the name `unnamed` by default. Currently there is no way to change this. The field after the date and time of the scan is a count of the number of hosts that were scanned. The next field is the username of the userid from which the scan was run.

```
2000/08/01 15:41:26    1 guenter    unnamed
2000/08/02 14:54:36    1 guenter    unnamed
2000/08/04 10:08:13    1 guenter    unnamed
2000/08/04 10:19:42    1 guenter    unnamed
2000/08/04 10:23:45    1 guenter    unnamed
2000/08/04 14:17:48    1 guenter    unnamed
2000/08/04 14:24:59    1 guenter    unnamed
```

Figure 46. Output from NSA index command

5.6.4.2 Merge multiple databases into one

The `merge` subcommand is used to create a *new* database that is built from another set of databases. All of the scan data from the other databases is used to create the new database. Note that one cannot insert scans into an existing database. The first argument to the `merge` subcommand is the name of the new database. NSA will terminate with an error if this database already exists. All of the remaining arguments are the names of databases to merge together. Example:

```
# nsa merge merged.nsa database.nsa db.new.nsa
```

5.6.5 Utilities

NSA comes with a set of additional tools to perform supporting tasks that do not require the full runtime environment of NSA. They are:

- `html2text`
- `htmlpretty`
- `nsamon`
- `nsacc`
- `nsadis`

Converting HTML files to text

The `html2text` utility can be used when a HTML report is generated but text output is desired with the same look as an NSA generated text report (actually the same conversion routines are used). Do not attempt to use this utility with HTML code that was not generated by NSA, as it only supports a subset of HTML. The syntax is simple:

```
html2text html-file [text-file]
```

If `text-file` is specified, then `html2text` formats the document straight into the file; otherwise the report is formatted in memory and displayed to the terminal.

Beautifying HTML documents
This may be of use if you are sending the raw HTML output of NSA to someone else, as the normal output is quite ragged. It can also be useful if you want to do some post edits to the HTML, as it makes it much easier to read. However, the command had no effect on the simple reports that we created during this residency using the default templates.

Monitoring a running NSA
The `nsamon` utility is used to access the NSA monitoring port enabled via the `--monitorport` command line option. There is only one parameter, `monitor-port-number`. It indicates the port number specified with the `--monitorport` option when the scan was started. The interface is identical to that available from the keyboard where NSA was invoked (see Section 5.4.1, "Hotkeys" on page 82) with the addition of the key sequence Ctrl-D, which terminates the monitoring session.

NSA script compiler
The `nsacc` utility is used to compile an NSA script file into a byte code. It is part of the scripting system

NSA byte-code disassembler
The `nsadis` utility displays the byte code instructions from the specified file. It serves as a debugging tool for self-written NSA scripts.

5.6.5.1 NSA scripting language
NSA provides its own scripting language to allow easy addition of new code. The syntax of the scripting language resembles C and PERL. NSA does not execute the scripting language directly. One must first compile the script into a byte code.

This is not only a way for advanced users to enhance the functionality of NSA, it can also be a way to respond quicker to new attack forms, vulnerabilities and exposures. It is far easier for IBM to distribute a new script than to generate a new version of NSA with the same code built into it.

5.7 Best practices

The following FAQ was generated by interviewing skilled users. The individuals labeled A1 offer scanning services, so they work on many different sites for a limited period of time. Those labeled A2 are in charge of a large corporate network, including server farms as well as networking infrastructure devices.

Q: Which control files do you change frequently?

A1: Scannerdata to update for vulnerability tests. (There is a local version of this.)

A2: We did our own scannerdefs. We use the standard scannerdata, but we have added a second one by using the `--localscandata` parameter. That is where we apply our local extras.

Q: Can you expand on those extras a little bit:

A2: We are running web server farms, so there is no way we can disallow HTTP on port 80. So we run all sorts of checks to make sure there is no vulnerability left. Our network devices from different vendors all have their specific user ids. I don't want to name them here.

And there is this huge networking project that went on for years throughout the company. People were working very hard to get everything done in time, lots of overtime, and they went through highly exposed critsits. You can imagine they could not be bothered with potential security issues like weak passwords and community names or open TFTP servers at that time. Once matters had calmed down we started pushing them towards tighter security. We knew the names and passwords they had been using all over the place. So we fed those into NSA to keep track of every single instance. The networking guys did not love us for this.

Q: Which ones did you change initially and never or hardly revisited?

A1: config is very rarely revisited, scanner rules is rarely changed (There is now a local version of this). Scannerdefs is changed moderately (There is also a local version of this).

A2: We sat down at the beginning and put together a list of ports we felt we had to scan regularly.

Q: Can you name those?

A2: For TCP: all the well-known ones, 1025, 1080, 1352, 1524-1527, 1661-1672, 1760, 1812, 2049, 2784, 3264, 6000-6063, 6666, 6667, 7000-7009, 8080, 32700-32800.

Q: What kind of reports do you produce?

A1: We use the standard report, sometimes the policy reports, and we post-process the raw reports. A sysadmin would use the standard and policy

reports more. The policy and also the delta reports are key because you'll see change. Also there is an option to produce separate reports for each host.

A2: Standard, summary, vulnerability mostly, and policy, of course, especially when we have an audit. We do lots of references. They impress our executives, and they are useful, too.

Q: What are the most frequently used scan types?

A1: It depends on your purpose. The philosophy should be to be as gentle as possible in testing. That means that tests for buffer overflows and the like have to be turned on explicitly in a controlled fashion. You want to ensure that you do the tests but may need to make special arrangements the first time you run them on a host. That's where you could use the hosts files to match scan types to hosts.

We use a lot of customized scan types. The defaults don’t fit all sizes. At some point you want to be more complete. That's why there are options to sample ranges of ports.

A2: We test a lot for selected vulnerabilities, we do the port scans that I touched on, we probe into TFTP and SNMP servers a lot.

Q: Where in the network would you place NSA?

A1: The limit is set by the IP restrictions. Therefore you want to have the mobility of a laptop in many cases. Do not run it against itself (for example don't put it on a firewall) on a regular basis. Also, be careful if you're outside a firewall that your machine is hardened. You can run NSA on itself to help confirm hardening but you'll see some listening ports that NSA opens to work with (netbios).

Most people will probably want to test firewalls with it. Run both inside and outside and compare. That will show you if the filters are working, and what you will see if the filters fail.

A2: We use both fixed installation and laptops. There is no general rule, really.

Q: Anything that gave you problems you found difficult to solve?

A1: Nothing that serious, but there are common problems that occur. Stuff like ordering of positional command line arguments. This is a concept that requires getting used to. I recommend using NSA host files to keep organized, especially in a repeated testing environment. Also, use the database options (a must if you do deltas).

Q: Name the 3 most useful ways to run NSA.

A1: Initial checks, policies, and deltas.

A2: Vulnerabilities, automated regular runs (via crontab), policy scans against the corporate policy.

Q: Thank you for your time.

Chapter 6. Integrating third party products into Risk Manager

This chapter describes ways to integrate other event sources with the Tivoli SecureWay Risk Manager product. Risk Manager is developed on top of the Tivoli Framework to allow for an extensible and expandable management framework that enables you to integrate a multitude of Intrusion Detection Systems and other sources of events.

In this chapter we explain the steps necessary to perform a custom integration without making use of the pre-adapters that come with the Risk Manager product.

We will show two examples of integration; one is the reporting of successful ftp logins on a specific UNIX host, while the other notifies Risk Manager of failed login attempts on an Oracle8 database.

The concepts used and referred to are Tivoli Framework related, and are only mentioned for completeness. Knowledge of Tivoli Framework concepts, use of command line options, and the GUI is a prerequisite.

6.1 Prerequisites

The Tivoli products required for use with Risk Manager include:

- Tivoli Management Enterprise Framework (formerly TME/10 Management Enterprise Framework), Version 3.6.2
- Tivoli Enterprise Console (TEC), Version 3.6.2
- Tivoli Management Agent endpoint software (formerly LCF endpoint), Version 3.6.2 (required only when a sensor is installed on a Tivoli endpoint)
- The TME adapters (logfile or Windows NT)—installed where the IDS sensor systems reside
- An external relational database management system (RDBMS) for use as the event database

The above Tivoli products are not provided as part of the Risk Manager product. This guide does not include instructions for installing these prerequisite and co-requisite Tivoli products. Instead, refer to the Tivoli documentation for the each product for installation instructions. Likewise, support for these products is not provided as part of customer support for Risk Manager.

For more information see chapter 1 of *Risk Manager Installation and User's Guide*.

6.2 Test environment

We used a small environment in which we were able to use logfile adapters for UNIX systems and event adapters for NT systems. We also integrated ISS RealSecure into our environment to be able to evaluate and describe the Risk Manager correlation component in Chapter 4, "Risk Manager correlation" on page 27. ISS RealSecure is not discussed in this redbook because it uses the Tivoli off-the-shelf Risk Management adapter. A comprehensive description on how to integrate ISS RealSecure, Tivoli WebIDS, and CISCO Secure IDS can be found in the Risk Manager User's Guide.

Furthermore we used the Network Services Auditor (NSA) for generating attack signatures and create a product description, the result of which can be found in Chapter 5, "Network Services Auditor (NSA)" on page 65.

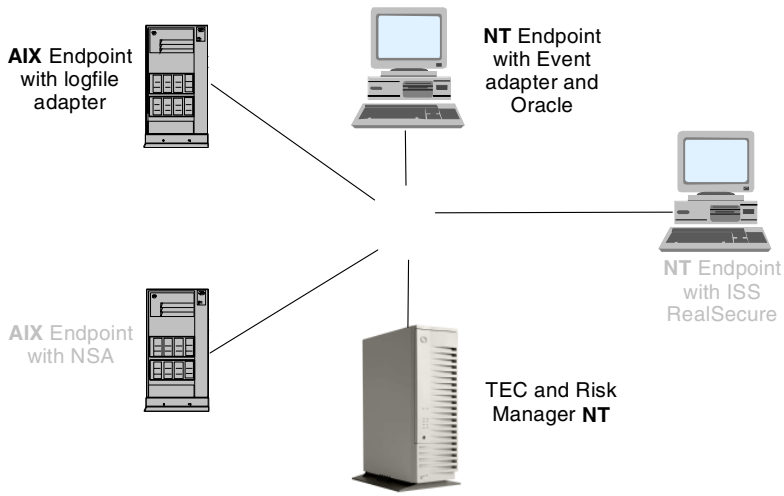


Figure 47. Tivoli SecureWay Risk Manager integration environment

We ran TEC 3.6.2, AIX, 4.3.3, NT 4.0, and Risk Manager 1.0.

6.3 Integration concepts

Integration of new sources for intrusion detection or security related events is done by creating or updating the adapters. An adapter is either a UNIX daemon or a Windows NT/Windows 2000 service that is capable of capturing

logfile events and formatting them for presentation to the TEC Server where the Risk Manager rule logic resides. Figure 48 is a high level flow of steps that need to be taken in order to integrate sensors into the Tivoli Enterprise Console (TEC) and to correlate events via the Risk Manager Correlation Component.

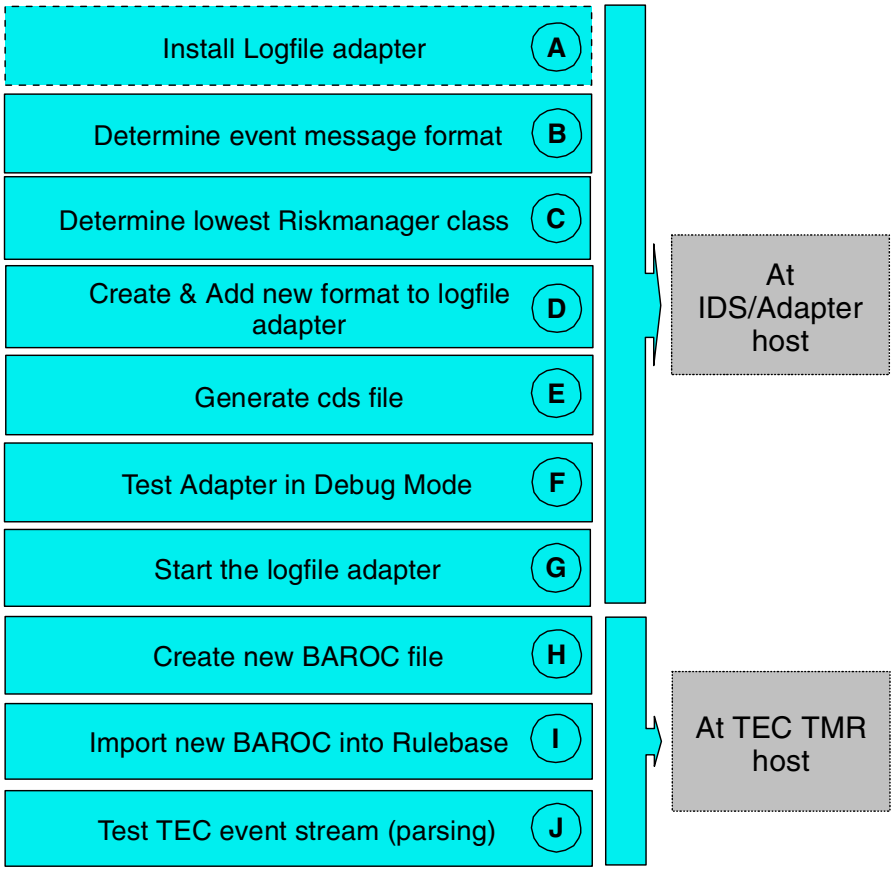


Figure 48. Event integration with Risk Manager - general steps

6.3.1 Integration steps

The boxes A through J in Figure 48 will be explained in greater detail in the following sections. Even though there are a lot of operating system specific and Tivoli specific details, the issues covered are mainly related to the Risk Manager Correlation Component and the TEC Server as opposed to issues related to the client or TEC adapter site. You will find a step by step walkthrough of the different activities and decisions that need to be made in order to create a Risk Manager adapter.

6.3.2 Install logfile adapter (A)

Before an event can be sent to TEC and the Risk Manager correlation component, the logfile adapters for NT and AIX have to be installed on the system where the logfile is being stored. We will discuss the NT logfile adapter and the UNIX logfile adapter in a bit more detail to understand their capabilities for integration purposes. These adapters are part of the Tivoli Framework. For more information on the installation and configuration of the adapters, refer to the *Tivoli Framework User's Guide*.

The logfile adapters are responsible for generating, formatting, and sending events to the TEC server. The adapters are also responsible for discarding unwanted events using local filtering to help reduce network traffic and the load on the event server.

The logfile adapters can exist on any TCP/IP connected system, not just on Tivoli managed nodes.

We will refer to the UNIX logfile adapter and the NT event adapter as event adapters throughout this chapter.

Note

For each event source an event adapter needs to be identified, for instance, to forward NT event log information to TEC, we need to install the Windows NT event log adapter.

6.3.2.1 Files associated with event adapters

The following list explains the contents of the files associated with the event adapter. Every adapter uses some of these files, but may not use all of them depending on the function it performs.

Global configuration file (.conf)	Tells the adapter where to find the event server, sets options such as maximum message size, and provides some filtering options.
Format configuration file (.fmt)	Defines the format of system log messages and their mappings to classes.
Adapter-specific configuration file (.cds)	This file is generated from the tecad_logfile.fmt file. The selection of messages by the logfile adapter is based on the specification of this file.

Event class definition file (.baroc)	The class definitions required by the TEC server to process the event. This file is imported into the TEC server as a class definition and compiled. Each adapter comes with pre-defined baroc definitions.
Error file (.err)	Contains error logging and tracing options for some adapters. The debugging information can be sent to a file or panel.
Rules file (.rls)	Defines the default rules for the adapter. This file is imported into the TEC server and compiled.

6.3.2.2 UNIX logfile adapter

UNIX provides two ways how an error or warning message can be logged; the function syslog or the error reporting program.

The syslog relies on a process called syslogd. This process will report operating system conditions to the system console or to a specific logfile.

Messages are tagged with codes indicating the classification of the message. A classification, known in UNIX as a priority, is encoded as a facility, which describes the component of the system generating the message, and, as a level, indicates the severity of the message.

The logger command, found in the /usr/bin directory, provides an easy-to-use interface to the syslog subroutine. A message variable can be specified on the command line, which is logged immediately, or a variable file is read and each line of the file variable file is logged.

The logfile adapter connects to the syslogd daemon by adding an entry to the file /etc/syslog.conf. This file controls the output of the syslog daemon. It supports subscriptions by other AIX processes to specific subsets of messages handled by the syslog daemon. Each entry in the file consists of two parts; a selector, which determines the messages of interest to the subscribing process, and a destination file to which the selected messages are directed. The selection of messages is based upon two criteria; the facility generating the message and the priority of the message. The facilities supported are:

- kern - Kernel
- user - User level
- mail - Mail subsystem

- daemon - System daemons
- auth - Security or authorization
- syslog - Syslogd daemon
- lpr - Line printer subsystem
- news - News subsystem
- uucp - UUCP subsystem
- * - (All facilities)

The priority levels that can be selected are:

- emerg - Emergency messages (LOG_EMERG) such as fixed-disk errors.
- alert - Important messages (LOG_ALERT) such as serious hardware errors.
- crit - Critical messages not classified as errors (LOG_CRIT), such as improper login attempts.
- err - Messages that represent error conditions (LOG_ERR), such as an unsuccessful disk write.
- warning - Messages for abnormal, but recoverable, conditions (LOG_WARNING).
- notice - Informational messages. All messages without a priority designation are mapped into this priority messages (LOG_NOTICE).
- info - Less significant informational messages (LOG_INFO).
- debug - Debugging messages (LOG_DEBUG).
- none - Excludes the selected facility.

Destinations may be specified as follows:

- /Filename - Full path name of file opened in append mode
- @Host - Host name
- User - User name(s)

The default entries appended to the /etc/syslog.conf created by the logfile adapter install process are shown below:

```
# Tivoli logfile adapter entry, ID=default
*.emerg;*.alert;*.crit;*.err;*.warning;*.notice;*.info  /tmp/.tivoli/.tecad_logf
ile.fifo.jupiter.itsc.austin.ibm.com.default.
```

Figure 49. syslog.conf entry

This entry directs the syslogd daemon to pass all messages to the specific file for the logfile adapter in /tmp.

The refresh command would normally be used to stop and restart syslogd after updating of the syslog.conf file. However, when the logfile adapter process is running the refresh command should not be used. Use the init.tecad_logfile script instead. In our case, it was located in:

```
/usr/local/Tivoli/bin/aix4-r1/TME/TEC/adapters/bin
```

There may be a performance benefit in tailoring the logfile adapter entry to indicate that only messages from specific facilities or message-specific priority levels get passed to the adapter. In addition, these entries should be reviewed to make sure that all of the messages you are expecting to be passed by the event adapter actually appear at the event server.

6.3.3 Determine event message format (B)

In this case we want to capture successful FTP logins on a specific host. The example message is an actual line created by syslogd on AIX:

```
Jul 26 12:50:51 jupiter ftpd[12488]: FTP LOGIN FROM venus.itsc.austin.ibm.com, root
Jul 26 12:51:05 jupiter ftpd[12488]: FTPD: EXPORT file local , remote bar.
```

Figure 50. Syslog entry for successful FTP login

It is assumed that a complete analysis of the security threats that might occur has been done. Be clear which events need to be captured before starting this exercise.

The format entry we added at the **end** in the adapter file *tecad_logfile.fmt* is a standard TEC format file approach, as is shown in the example below.

```
//Jul 29 15:27:46 jupiter ftpd[14804]: FTP LOGIN FROM mars.itsc.austin.ibm.com,
//root
FORMAT FTPsuccesslogin
%t %s ftpd[%s]: FTP LOGIN FROM %s %s
END
```

Figure 51. Example format file for successful FTP login

It is considered good practice to show the string we want to capture before the FORMAT statement for documentation purposes.

- %t is a special placeholder for date/time strings.
- s% is a regular placeholder for all regular strings.

Once we know what information to capture, we need to map the information to a Risk Manager class to enable Risk Manager correlation component processing as explained in Section 6.3.5, “Add and create new format for logfile adapter (D)” on page 140.

6.3.4 Risk Manager class hierarchy (C)

This is a critical part because the Risk Manager correlation component depends on this to correlate the event feed from the event adapters and Intrusion Detection Systems.

To better understand what information needs to be mapped to certain Risk Manager classes, look at Figure 52 and Figure 53 on page 139. Risk Manager base classes are defined in *idwg.baroc* file and should be used as a base class for generic events related to network-based IDS, host-based IDS, routers, firewalls, etc., specifically designed to facilitate rapid development of an adapter. This file is only used as a skeleton, and is not part of the active environment as described in Chapter 4, “Risk Manager correlation” on page 27. An adapter developer could use this class initially before developing a BAROC class file specifically for the sensor.

Please note that only relevant classes for integration with custom build adapters have been listed.

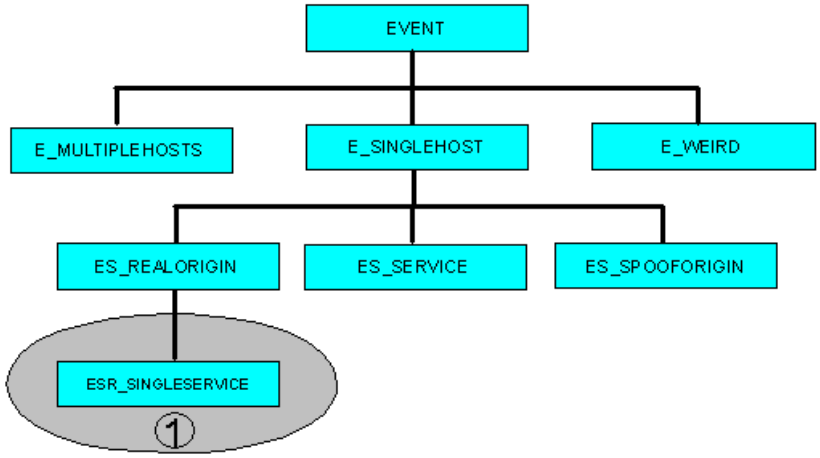


Figure 52. Class hierarchy for base level classes

An explanation of the main classes shown in Figure 52 is given below.

EVENT

The event class is the top level class for Risk Manager classes, and is referred to as IDS_Event. IDS_Event is the ancestor of all the Risk Manager classes.

Classes that derive from IDS_Event are used for events coming from security sensors. Any security sensor that feeds information into the Risk Manager correlation component has a set of classes associated with it that ultimately derive from IDS_Event. The definition of new classes for a sensor type are critical. Careful consideration should be given to how the sensor events fit within the IDS_Event class structure.

Note

By deriving from base classes as low as possible in the tree, the new class provides as much specific relevant information as possible during the correlation process.

E_MULTIPLEHOSTS

This class is intended to carry more complex alerts that concern several targets in a single domain. An example of such an alert would be a system wide scan, where one source tries to connect to the same service on different targets. Multiple host destination (or target) identifications, which are involved

in this alert, are provided by sub classing. An example would be a wide port scan using TCP or UDP.

E_SINGLEHOST
For all custom build adapters the E_SINGLEHOST class should be used. This class is used to define custom build adapters that are single host related.

Most simple alerts coming from intrusion detection sensors are expected to be in this hierarchy. The first component being highlighted in the alert is the destination/target of the alert. The destination of the attack is either the recipient of the action in the case of a remote attack (e.g. destination of the IP packet, host name to which the attacker logs in), or the host on which the action is carried out in the case of a local attack. Remote attacks for which the intrusion detection system has no information concerning the remote host should be reported at this level or using the AS_APPLICATION subclass hierarchy.

E_WEIRD
Contains internal messages.

ES_REALORIGIN
The validity of the source host information can not be determined from the nature of the signature.

This class contains alerts when there is no way to differentiate between a spoofed origin and the real source of the alert. If it is possible to detect a spoofed IP address, the ES_SPOOFORIGIN class is used.

Intrusion Detection Systems give no guarantee that the source given in the alert is the one that actually carried out the attack; therefore most events are classified under the ES_REALORIGIN class.

ES_SPOOFORIGIN
From the nature of the signature, the source host information is known to have been falsified.

Some attacks require spoofing the origin of the packet. This set of subclasses contain alerts for which we are certain, that the source address is not the origin of the alert. A typical example of this class of alerts is the Land attack, when source and destination addresses are the same. When it is not possible to determine if the address is false, then the ES_REALORIGIN subclass hierarchy is used (even though the address may still be wrong).

ESR_SINGLESERVICE

This class covers all intrusion detection alerts that contain the triplet (destination, source, service). As such, most intrusion detection alerts will be mapped into subclasses of this class.

This is the class we will use in our integration examples.

Figure 53 shows the sub class hierarchy of ESR_SINGLESERVICE. Sub classes specific to off-the-shelf intrusion detection adapters have been omitted from this chart.

A more complete list of the most widely used classes and their attributes is listed in Appendix A, “Risk Manager class description and attribute list” on page 167.

.

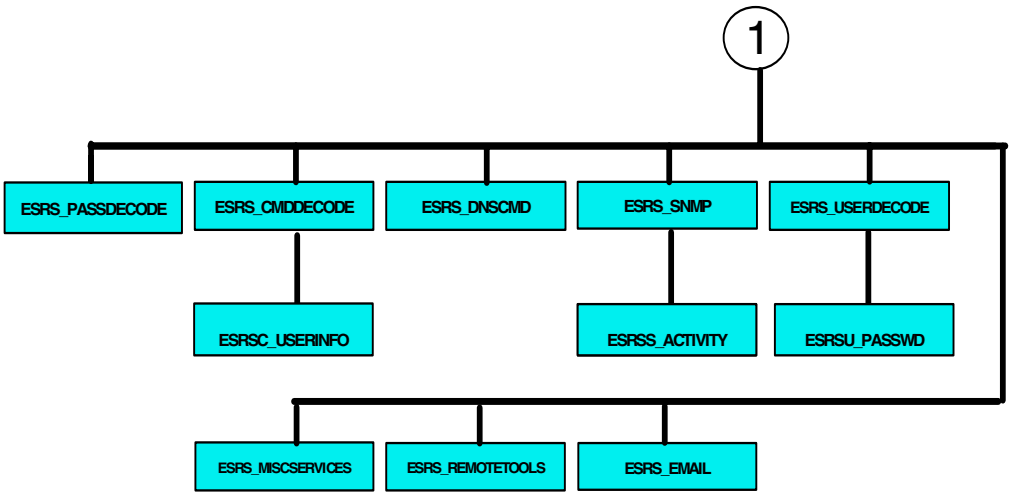


Figure 53. ESR_SINGLESERVICE class hierarchy

How to select a base class

Whatever the selection will be, just keep in mind that you will:

- Only use base classes defined in the *ids_abstract.baroc*
- Derive all new classes from a class in the tree described in the *ids_abstract.baroc*

Refer back to the example of the ftp login in Section 6.3.3, “Determine event message format (B)” on page 135. This example will be used to show the decision process and how to get to the lowest possible base class.

Note

The lower the base class in the ids_abstract.baroc hierarchy, the better the correlation result will be.

- IDS_EVENT: the starting point, IDS_EVENT is the ancestor of our class
- E_SINGLEHOST: the event involves a single destination host
- ES_REALORIGIN: the event involves a single source host, and there is no certainty that the source host name is spoofed
  - Note that ES_REALORIGIN versus ES_SPOOFEDORIGIN might be somewhat misleading since ES_REALORIGIN does not imply that the source name is not spoofed.

Note

The classname ES_REALORIGIN might change in a next release of the Risk Manager product. Although the name implies a verified real origin address, it only shows us addresses that are not recognized as spoofed addresses but still have a possibility of being spoofed after all.

- Next we select ESR_SINGLESERVICE: this event involves the FTP service
- ESRS_USERDECODE: a user related event
  - If a password had been available, the ESRS_USERDECODE would have been the lowest class level.

6.3.5 Add and create new format for logfile adapter (D)

Now we have defined our mapping we can finish our format file that will be used by the logfile adapter.

```
//Jul 29 15:27:46 jupiter ftpd[14804]: FTP LOGIN FROM mars.itsc.austin.ibm.com,
//root
FORMAT FTPsuccesslogin
%t %s ftpd[%s]: FTP LOGIN FROM %s %s
msg PRINTF("FTP (Ton) LOGIN BY %s FROM %s TO %s", esrsu_user, esr_src_hostname,
 e_idsid_hostname)
date $1
e_idsid_hostname $2
es_dst_hostname $2
esr_src_hostname $4
esrsu_user $5
e_date $1
hostname $2
e_idsid_process $3
e_signature PRINTF("FTP (Ton) LOGIN BY %s FROM %s TO %s", esrsu_user, esr_src_hostname
e_idsid_hostname)
END
```

Figure 54. Example mapped to ids_abstract.baroc format file

A list of base classes and their attributes can be found in Appendix A, “Risk Manager class description and attribute list” on page 167.

The FORMAT statement defines the message as consisting of an arbitrary number of strings, with the date and time stamp in the first position followed by hostname and an arbitrary number of strings in the message body.

The format statement for FTPsuccesslogin then further specializes the mapping to search for the string ftpd in the third position of the message. If found, it assigns the value of ftpd to the subsource event slot, the fourth position is the process id (pid).

Finally, the format statement for FTPsuccesslogin refines the search criteria to include the string “FTP LOGIN” beginning in string position five. This will ensure that all FTP logins on this host that are successful will be selected by the newly created adapter and sent to the Tivoli Enterprise Console for processing by the Risk Manager correlation component.

Note

This example illustrates the ease and flexibility of the adapter format file. Virtually any application that is able to generate security related output in the form of a log file can be integrated into Risk Manager.

The following fields are required in this example for the Risk Manager correlation component to work:

- **e_date** The UNIX epoch time (the number of seconds since Jan 1, 1970 00:00:00.)
- **e_idsid_hostname** The name of the host the adapter is running on
- **es_dst_hostname** The destination hostname of the activity
- **es_src_hostname** The hostname of the machine the activity originated on
- **es_src_haddress** If the IP address is known
- **e_signature** This field will be used in the aggregation process to identify security situations
- **e_esrsu_user** Will hold the username
- **e_idsid_process** The process id of the FTP action

The next three fields are found functional and should be included in the format file:

- **date** A standard TEC attribute
- **msg** A standard TEC attribute that allows the message to show up in the Risk Manager event console
- **hostname** A standard TEC attribute that allows the hostname to show up in the Risk Manager event console; if omitted, this field displays a blank

Note

e_date expects the UNIX epoch time (the number of seconds since Jan 1, 1970 00:00:00.). Tivoli does not provide a way to convert this with the event adapter. However, a rule can be created to work around this issue. For a detailed explanation see Section 6.5.2, “e_date conversion” on page 164.

Logfile adapter configuration files

We want to mention some brief comments on the logfile adapter configuration files. For more detailed information about these logfiles, please refer to the *Tivoli Event Console Adapters Guide.*

tecad_logfile.conf

The section in the *tecad_logfile.conf* most important to update is the filter section on the lower part of the file.

This section specifies how events are filtered. When the logfile adapter is installed, the configuration file already has four filter statements. These filter statements for Logfile_Base, Logfile_Sendmail, Amd_Unmounted, and Amd_Mounted would block any events that match the base class for these event classes (these are messages for which there is no specific event class in the .baroc file).

The entries in the *tecad_logfile.conf* file are similar to those listed below.

```
Filter:Class=Logfile_Base
Filter:Class=Logfile_Sendmail
Filter:Class=Amd_Unmounted
Filter:Class=Amd_Mounted
```

It is also possible to add more logfile sources by adding lines to the tecad_logfile.conf and prefixing them with 'LogSources' as shown in the example below.

```
LogSources=/tmp/applog
```

tecad_logfile.fmt

The tecad_logfile.fmt resides on the same system where the logfile adapter is running. When new sub classes or classes are added to this file, a new cds file has to be generated and the BAROC file, which resides on the TMR server, needs to be updated with the new classes.

6.3.6 Generate cds file (E)

The selection of messages by the logfile adapter is based on specifications provided in the tecad_logfile.cds file. This file is generated from the FORMAT statements defined by the user in the tecad_logfile.fmt file by the logfile_gencds utility.

The fmt file is located in /usr/local/Tivoli/bin/aix4-r1/TME/TEC/adapters/etc, and the gencds utility in /usr/local/Tivoli/bin/aix4-r1/TME/TEC/adapters/bin. The utility is executed by performing the following command:

```
./logfile_gencds tecad_logfile.fmt > tecad_logfile.cds
```

A sample output of the tecad_logfile.cds entries generated by the logfile_gencds utility for FTPsuccesslogin is shown below.

```
CLASS FTPsuccesslogin
  SELECT
    1: ATTR(=,"_event_id"), VALUE(=,"82");
    2: ATTR(=,"date");
    3: ATTR(=,"e_idsid_hostname");
    4: ATTR(=,"es_dst_hostname");
    5: ATTR(=,"esr_src_hostname");
    6: ATTR(=,"esrsu_user");
    7: ATTR(=,"e_date");
    8: ATTR(=,"hostname");
    9: ATTR(=,"e_idsid_process");
   MAP
    msg = PRINTF("FTP (Ton) LOGIN BY %s FROM %s TO %s", $V6, $V5, $V3);
    date = $V2;
    e_idsid_hostname = $V3;
    es_dst_hostname = $V4;
    esr_src_hostname = $V5;
    esrsu_user = $V6;
```

Figure 55. logfile_gencds output

6.3.7 Test adapter in debug mode (F)

This step enables us to see if the values we have assigned to the '$" slots are what we want them to be. This is always good practice because the next step will be the creation of the BAROC file. Ensuring that the adapter on the host is working properly allows us to concentrate on the TMR server and the Risk Manager correlation component.

For debugging:

1. Make sure that the logfile adapter instance is not running
2. Start the logfile adapter with the following parameter -d,

   ```
   ./init.tecad_logfile -d start syslog
   ```

This will start the logfile adapter and refresh the syslog daemon, and will display the adapter file on the screen then it waits for events that match one or more strings in the format file, in this case *tecad_logfile.fmt.*

A sample output is listed below after doing an ftp login to the host.

```
matched FTPsuccesslogin
        $1 is `Aug 2 11:30:32'
        $2 is `jupiter'
        $3 is `14162'
        $4 is `9.3.240.135,'
        $5 is `root'
```

Figure 56. Sample debug output

If the output is satisfactory, the event adapter can be put into production after the BAROC file(s) has/have been created as shown in step H.

To start the adapter in normal mode the following command has to be issued:

```
./init.tecad_logfile start syslog
```

This will refresh the syslogd and initialize the adapter for operation.

6.3.8 Create new BAROC file (H)

It is recommended to always create new BAROC files for new adapters and leave the standard BAROC files that come with the Risk Manager product untouched because you may want to go back to the default configuration.

The BAROC files are situated on the TMR. The BAROC file in this case is called FTPsuccesslogin.baroc.

```
TEC_CLASS:
 FTPsuccesslogin ISA ESRS_UserDecode
 DEFINES {
         e_idsid_tool: default = "LOGFILE";
         e_method: default = "knowledge";
         e_level: default=1.0;
         esrs_svcname: default = "ftp";
         esrsu_user: default ="none";
         severity: default ="CRITICAL";
};
END
```

Figure 57. FTPSuccess BAROC file

The BAROC file FTPsuccess is comprised of the following:

- TEC_CLASS is the label used to define a new event followed by the new event class name FTPsuccesslogin.
- FTPsuccesslogin is the event class name, which is really a subclass of the Risk Manager event (the base event class is defined in $BINDIR/TME/TEC/default_rb/TEC_CLASSES/riskmgr.baroc).

- ISA defines a superclass from which the new events inherit the base class EVENT (from riskmgr.baroc).
- DEFINES - keyword used to define new slot values for the event and to set default values for the slots.
- The slot names e_idsid_tool, e_method, e_level, and so on are the slots defined in the event class.
- The default value is the slot facet that is used to define the type of information contained in the slot. The default value is used when the slot value is not assigned by the adapter. In this example, the slot names have default values assigned, but it is not mandatory to do so. It does, however, improve readability and understanding of the defined CLASS.

The following fields need to be set in the FTPsuccesslogin.baroc file for successful event correlation:

- **e_idsid_tool** This will be set to the name of the sensor type, which in this case we have classified as LOGFILE. This entry is used to identify the different intrusion detection adapters. This helps to identify new adapters on the network.
- **e_method** Will always be set to KNOWLEDGE.
- **e_level** This is a numerical value. The default value is 1.0. This value is used in the weighing or importance determination relative to other events. For more information on this, please see Chapter 4, “Risk Manager correlation” on page 27.
- **esrs_svcname** This attribute of ESRS_UserDecode is set to ftp to identify the service name for correlation purposes.
- **esrsu_user** This attribute of ESRS_UserDecode is set to NONE because this will be set by the tecad_logfile.fmt mappings when a username is available. If a username is not available, esrsu_user will remain NONE.
- **severity** A TEC attribute used to set the event severity on the TEC and Risk Manager container entries.

Note

The e_level attribute is a crucial attribute of the Risk Manager correlation component. Its value is related to the threshold settings defined in riskmgr_config.pro

For more information on BAROC files, see the *Tivoli Enterprise Console User's Guide*.

6.3.9 Import new BAROC file (I)

The newly created FTPsuccesslogin BAROC file now has to be imported in the active Risk Manager rule base.

The following commands have to be executed in their respective order:

```
wimprbclass FTPsuccesslogin.baroc <rulebase>
```

Importing the new or modified FTPsuccesslogin.baroc into the existing rulebase <rulebase>.

```
wcomprules -t <rulebase>
```

This command will compile all the BAROC files in the rulebase.

Now the TEC server has to be stopped in order for the changes to take effect. This can be done either via the command line or the GUI interface on the Tivoli Enterprise Console.

The command line format is:

```
wstopesvr
sleep 10
wstartesrv
```

If modifications have been made to the FTPsuccesslogin.baroc after importing, you need to remove the BAROC file by using the following command:

```
wdelrbclass FTPsuccesslogin.baroc <rulebase>
```

This will unload the FTPsuccesslogin.baroc file from the rulebase <rulebase>. Then proceed with the import of the new BAROC file and compiling of the BAROC file(s) as shown at the beginning of this section.

6.3.10 Test TEC event stream (J)

Once integration is complete, the final step is to test if the events that are forwarded by the newly created event adapter FTPsuccesslogin are received and parsed successfully by the Tivoli Enterprise Console engine.

The following commands can be used to verify the reception of events in the event repository and reception log:

To show events in the event repository in descending order (newest first), issue the following command:

```
wtdumper -o DESC
```

```
0~0~ES~1~965305681(Aug 03 07:28:01 2000)~1~FTPsuccesslogin~
LOGFILE~SECURITY_LAGZUR~~N/A~jupiter~N/A~OPEN~
~[ admin]~CRITICAL~
Aug 3 07:24:12~
FTP (Ton) LOGIN BY root FROM 9.3.240.135, TO jupiter~
none~0~
```

Figure 58. wtdumper output

To list events received by the reception log using the newly created FTPsuccesslogin.baroc file to parse the events and prepare them for reception by the Tivoli Enterprise Console and subsequent Risk Manager correlation component processing, use the following command:

```
wtdumprl -o DESC
```

The output is shown in Figure 59.

```
### EVENT ###
FTPsuccesslogin;msg='FTP (Ton) LOGIN BY root FROM 9.3.240.135, TO jupiter';date
'Aug 3 07:24:12';e_idsid_hostname=jupiter;es_dst_hostname=jupiter;esr_src_hostn
me='9.3.240.135,';esrsu_user=root;e_date='Aug 3 07:24:12';hostname=jupiter;e_id
id_process=3328;e_signature='FTP (Ton) LOGIN BY root FROM 9.3.240.135, TO jupit
r';END

### END EVENT ###
PROCESSED
```

Figure 59. wtdumprl output

The PROCESSED statement at the end of this event indicated a successful acceptance by TEC and that the event has been submitted to the event reception log. As a result a successful parse will have a created entry in the RiskMgr_all container as shown in Figure 60 on page 149.

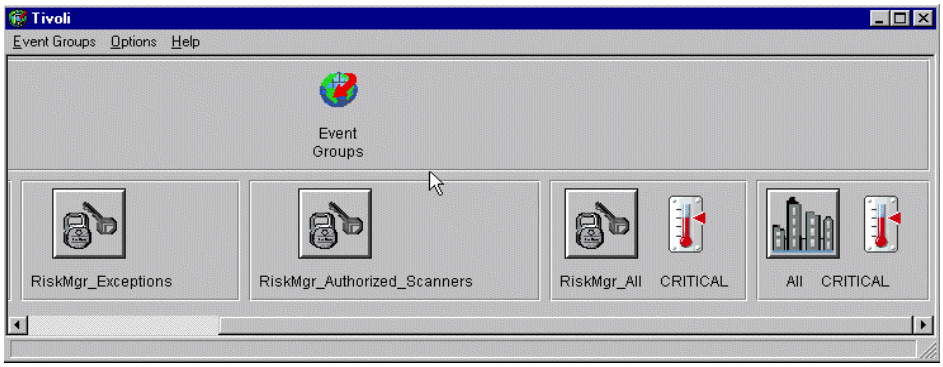


Figure 60. Risk Manager containers after FTPsuccesslogin reception

Figure 61 shows the Risk Manager RiskMgr_All container with the confirmation of the FTPsuccesslogin event reception.

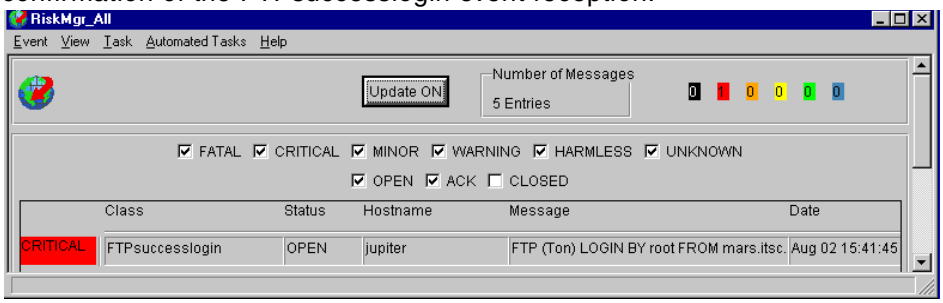


Figure 61. RiskMgr_All container view

6.4 Integration with Oracle

This section describes the steps involved to enable security related events from an Oracle8i database to be sent to the Tivoli Enterprise Console and be processed by Risk Manager's correlation component.

We will follow the same steps as outlined in "Integration concepts" on page 130 of this chapter.

The Oracle8i RDBMS is running on a Windows NT 4.0 Server with an installed Tivoli Endpoint.

In this example we will pass events to the NT event log. On UNIX platforms this can be done by piping this to the syslog file as described in Section 6.3.2, "Install logfile adapter (A)" on page 132.

6.4.1 Prerequisite

- Tivoli NT event adapter is installed on the NT Endpoint.
- Oracle 8.x is installed on the same host as the NT event adapter.

6.4.2 Integration concepts

Please see Figure 48 on page 131 for the sequential order of steps to send security related Oracle events to the Tivoli Enterprise Console.

Throughout this section we will refer to Figure 48 on page 131 by means of (A) for Install Event Adapter and so on.

The high level architecture is exactly the same as shown in Chapter 3, “Risk Manager architecture” on page 19; we now replace “other application” as shown in Figure 5 on page 20 with Oracle and “pre-adapter” with Oracle Audit trail. This is how we visualize the concept of integrating Oracle with TEC and Risk Manager. Figure 62 on page 151 represents the updated model for Oracle.

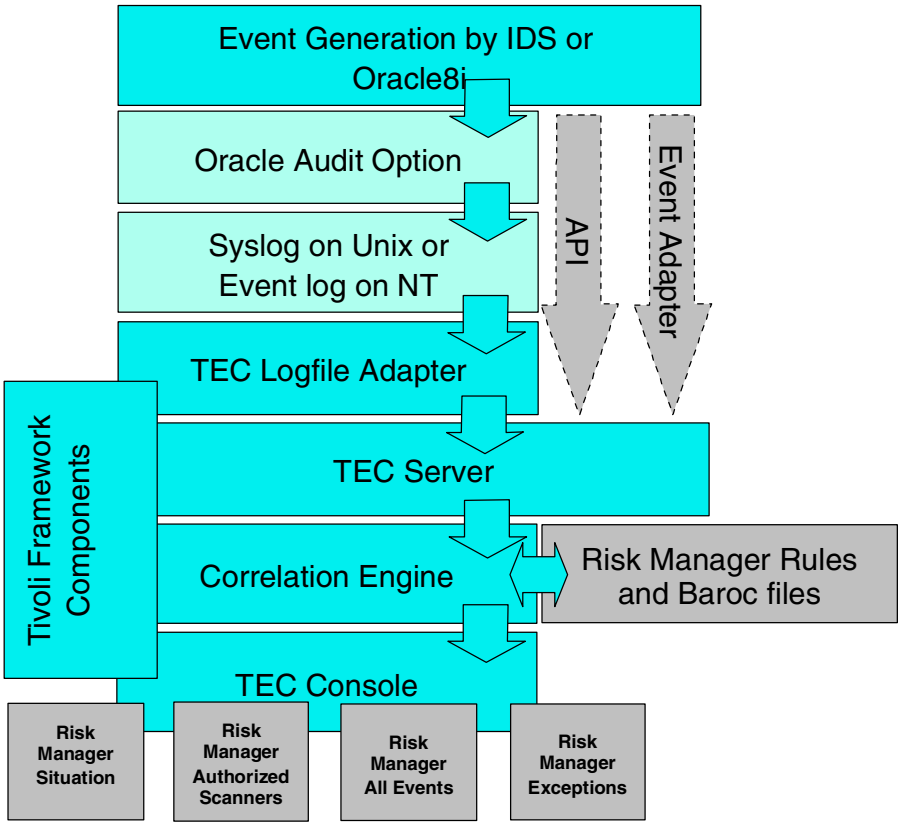


Figure 62. High Level Risk Manager overview with Oracle components

6.4.3 Why audit database logins?

A user may want to “guess” his way into a system. Auditing is a way of helping you detect when someone is trying to break into your database.

If a policy has been established in your company that uses a very specific username convention and that convention is widely known, guessing usernames may be a very easy task.

For most of us, passwords are made up of things we like or do, for example, children’s names, hobbies, months, birthdays, and so on.

Remember that we all make mistakes typing our password, and in a mid or large size organization these failed login attempts might reach hundreds a day if not more. The Risk Manager correlation component makes real attempts easier to detect.

6.4.3.1 Oracle INIT.ORA settings

In order for Oracle8 to write audit trail records to a file outside the database structure, the following parameters have to be set in the Oracle environment:

- Set the *audit_trail* parameter in the INIT.ORA file to:

  ```
  audit_trail = OS
  ```

- An optional setting is the *audit_trail_dest* parameter, which can be used to direct the audit trail to be written to a user specified file. The parameter syntax is:

  ```
  audit_trail_dest = <destination>
  ```

  In this case we do not set this parameter because we want to use the NT event log as output. UNIX platforms might want to use this option to create an output file.

  After editing the parameters in the INIT.ORA file, the Oracle instance must be restarted so that the INIT.ORA changes take effect.

6.4.3.2 Oracle RDBMS setting

Having changed the INIT.ORA parameters, the RDBMS has to be instructed to start auditing LOGON events. This can be done by issuing the following command in *isql*:

- To audit whenever a connection to the database was attempted but failed to logon, use:

  ```
  SQL>audit connect whenever not successful;
  ```

  This will write every failed login attempt to the event log as shown in Figure 63 and Figure 64 on page 153. For more details on the Oracle Audit command, please refer to the appropriate Oracle manuals.

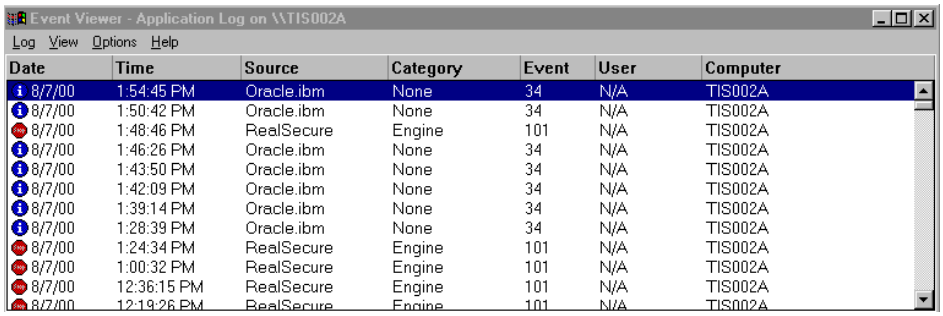


Figure 63. Snapshot of NT Event log with Oracle and RealSecure IDS events.

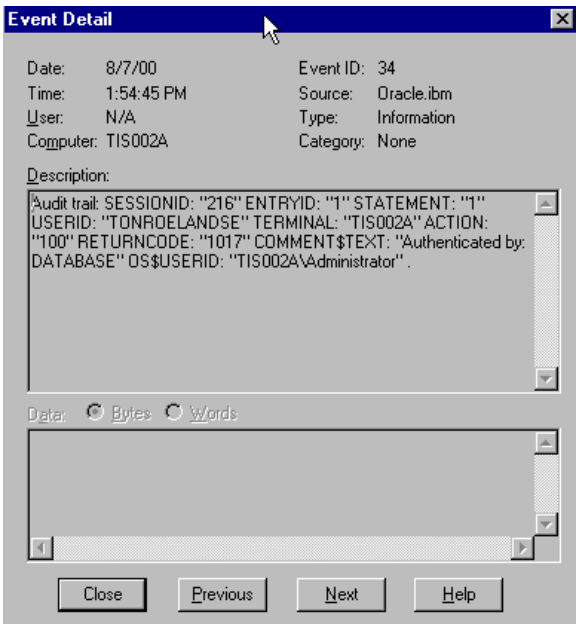


Figure 64. NT Event log reception of attempted Oracle login

6.4.4 Install NT logfile adapter

The installation of the NT logfile adapter is not discussed in detail in this section. Please refer the *Tivoli Adapters User Guide* for more details on this matter.

6.4.5 Determine event message format for Oracle login failure

To determine the information we want to capture we need to dissect the string sent to the Windows event log.

We have used the base format file for NT, tecad_nt.fmt, for NT event log based information, and have appended our format section to this file. This file makes a good base for NT events.

The following string contains all the necessary information to build a Risk Manager event.

```
//Audit trail: SESSIONID: "168" ENTRYID: "1" STATEMENT: "1"
//USERID: "TON" TERMINAL: "TIS002A" ACTION: "100" RETURNCODE: "1017"
//COMMENT$TEXT: "Authenticated by: DATABASE" OS$USERID: "TIS002A\Administrator"
FORMAT OR_LoginFailure
%t %s %s %s %s %s %s Audit trail: %s %s %s %s %s %s %s %s %s %s %s %s %s "1017" %s
%s %s*
```

Figure 65. NT event format for Oracle login failure

The structure of the NT event log differs from the UNIX syslog format. Therefore, the FORMAT string should always start with:

```
%t %s %s %s %s %s %s <name of event followed by :>
```

This is specific to the NT event log file format.

The format string in our example has the following elements:

```
%t %s %s %s %s %s %s Audit trail: %s %s %s %s %s %s %s %s %s %s %s %s %s
"1017" %s %s %s*
```

Where “Audit trail” and “1017”, which is the Oracle return code for wrong password/login, are the keywords that act as a filter and ensure that only events with a combination of these two strings will be sent to the Tivoli Enterprise Console and processed by the Risk Manager correlation component.

Note

Auditing can be done on anything from table drops and creation to record inserts and updates by individual users. In this example we just looked at login failures. There are over 80 user generated audit events in Oracle available that can be captured and used with Risk Manager.

6.4.6 Determine Risk Manager class mapping for Oracle events

For determination on how to map fields to Risk Manager classes, see Section 6.3.4, “Risk Manager class hierarchy (C)” on page 136.

Again in this case, and in most other cases where you have to deal with a user generated event, ERSR_userdecode has been used, which is a subclass from the ESR_Singleservice class definition.

The completed format file is shown below:

```
//Oracle  extension by Ton Roelandse for IBM Redbook "Tivoli Risk Manager v1.0"
//Audit trail: SESSIONID: "168" ENTRYID: "1" STATEMENT: "1"
//USERID: "TON" TERMINAL: "TIS002A" ACTION: "100" RETURNCODE: "1017"
//COMMENT$TEXT: "Authenticated by: DATABASE" OS$USERID: "TIS002A\Administrator" .
FORMAT OR_LoginFailure
%t %s %s %s %s %s %s Audit trail: %s %s %s %s %s %s %s %s %s %s %s %s %s "1017" %s %s %s*
msg PRINTF("SOURCE: %s FAILED LOGIN ATTEMPT BY %s, FROM %s, ACTION %s, RTNCODE 1017",
hostname, esrsu_user, origin, actn)
hostname DEFAULT
origin DEFAULT
sub_source $6
-actn $19
-date1 $1
-date2 $2
date PRINTF("%s %s", date1, date2)
sub_origin PRINTF("%s",hostname)
e_date PRINTF("%s %s", date1, date2)
es_dst_haddress PRINTF("%s", hostname)
esr_src_haddress PRINTF("%s", origin)
e_idsid_hostname $11
esrsu_user $15
e_signature PRINTF("SOURCE: %s FAILED LOGIN ATTEMPT BY %s, FROM %s, ACTION %s, RTNCODE
1017", hostname, esrsu_user, origin, actn)
END
```

Figure 66. Oracle login format file

The following is a list of mandatory attributes used in this FORMAT file example:

- **e_date** An integer that uses the UNIX epoch time (the number of seconds since Jan 1, 1970 00:00:00.)
- **e_idsid_hostname** The name of the host the adapter is running on.
- **es_dst_hostname** The destination hostname of the activity.
- **es_src_hostname** The hostname of the machine the activity originated or
- **es_src_haddress** If the IP address is known.
- **e_signature** This field will be used in the aggregation process to identify security situations.
- **esrsu_user** Contains the username of the attempted login session.

The next four fields are found functional and should be included in the format file:

- **date** A standard TEC attribute.

- **msg** — A standard TEC attribute that allows the message to show up in the Risk Manager event console.
- **hostname** — A standard TEC attribute that allows the hostname to show up in the Risk Manager event console. If omitted this field displays a blank.
- **origin** — A standard TEC attribute that contains the name or address of the originator of the event.

Note

Fields in a format file prefixed by a '-', like -act are temporary values and are used to build PRINTF statements. Fields like -act are not defined anywhere else, and only contain a value when the adapter executes the lines between FORMAT and END. The variables will be empty outside the Format - END area.

6.4.7 Generate CDS file for Oracle format file

Once the format file is completed it has to be compiled in order to generate the .cds file.

To create the .cds file run:

```
nt_gencds tecad_nt.fmt > tecad_nt.cds
```

If the generation of the .cds file finishes without errors, the NT event adapter can now be started to capture the NT events and forward them to the Tivoli Enterprise Console. The tecad_nt.cds entries for the new OR_LoginFailure class are shown in the screen sample below.

```
CLASS OR_LoginFailure
  SELECT
    1: ATTR(=,"_event_id"), VALUE(=,"220");
    2: ATTR(=,"hostname");
    3: ATTR(=,"origin");
    4: ATTR(=,"sub_source");
    5: ATTR(=,"actn");
    6: ATTR(=,"date1");
    7: ATTR(=,"date2");
    8: ATTR(=,"e_date");
    9: ATTR(=,"e_idsid_hostname");
    10: ATTR(=,"esrsu_user");
  MAP
    msg = PRINTF("SOURCE: %s FAILED LOGIN ATTEMPT BY %s, FROM %s, ACTION %s, RTNCODE
1017", $V2, $V10, $V3, $V5);
    hostname = $V2;
    origin = $V3;
    sub_source = $V4;
    date = PRINTF("%s %s", $V6, $V7);
    sub_origin = PRINTF("%s", $V2);
    e_date = PRINTF("%s %s", $V6, $V7);
    es_dst_haddress = PRINTF("%s", $V2);
    esr_src_haddress = PRINTF("%s", $V3);
    e_idsid_hostname = $V9;
    esrsu_user = $V10;
    e_signature = PRINTF("SOURCE: %s FAILED LOGIN ATTEMPT BY %s, FROM %s, ACTION %s,
RTNCODE 1017", $V2, $V10, $V3, $V5);
END
```

Figure 67. OR_LoginFailure cds file

6.4.8 Test Oracle adapter in debug mode

Once the .cds file has been created, it is good practice to run the tecad_nt -d option to run the adapter in debug mode to test the assignments of variables. Use the tecad_nt -d command in the ..\TEC\TME\adapters\bin directory. Before executing this command, please make sure that the TECNTAdapter service is not running.

```
tecad_nt -d -c c:\Tivoli\bin\w32-ix86\TME\TEC\adapters\etc\tecad_nt.conf
```

This allows us to test the following behaviors and outcome:

- The adapter responds to information.
- The format file we have created is capturing the intended message.
- If the slot assignment for the variables are acceptable. See output of tecad_nt below.

```
eventString=Aug 07 16:26:48 2000 0 Information N/A Oracle.ibm 34
Audit trail: SESSIONID: "168" ENTRYID: "1" STATEMENT: "1"
USERID: "TON" TERMINAL: "TIS002A" ACTION: "100" RETURNCODE: "1017"
COMMENT$TEXT: "Authenticated by: DATABASE" OS$USERID: "TIS002A\Administrator"
matched OR_LoginFailure
        $1 is `Aug 07 16:26:48'
        $2 is `2000'
        $3 is `0'
        $4 is `Information'
        $5 is `N/A'
        $6 is `Oracle.ibm'
        $7 is `34'
        $8 is `SESSIONID:'
        $9 is `"168"'
        $10 is `ENTRYID:'
        $11 is `"1"'
        $12 is `STATEMENT:'
        $13 is `"1"'
        $14 is `USERID:'
        $15 is `"TONROELANDSE"'
        $16 is `TERMINAL:'
        $17 is `"TIS002A"'
        $18 is `ACTION:'
        $19 is `"100"'
        $20 is `RETURNCODE:'
        $21 is `COMMENT$TEXT:'
        $22 is `"Authenticated'
        $23 is `by: DATABASE" OS$USERID: "TIS002A\Administrator" .'
```

Figure 68. Debug screen from tecad_nt

The debug can be cancelled by pressing CTRL-C.

The NT event adapter runs as a service, and can be stopped with the following command line:

```
net stop TECNTAdapter
```

and started with:

```
net start TECNTAdapter
```

6.4.9 Baroc file for Oracle

The BAROC file we have created for the OR_LoginFailure class is shown below:

```
TEC_CLASS:
 OR_LoginFailure ISA ESRS_UserDecode
  DEFINES {
         e_idsid_tool: default = "NT_Base";
         e_method: default = "knowledge";
         e_level: default=1.0;
         esrs_svcname: default = "oracle";
         esrsu_user: default ="none";
};
END
```

Figure 69. OR_LoginFailure BAROC file

The BAROC file for OR_LoginFailure is comprised of the following:

- TEC_CLASS is the label used to define a new event followed by the new event class name OR_LoginFailure.
- The OR_LoginFailure is the event class name, which is really a subclass of the Risk Manager top class event (the base event class ($BINDIR/TME/TEC/default_rb/TEC_CLASSES/riskmgr.baroc).
- ISA defines a superclass from which the new events inherits base class EVENT (from riskmgr.baroc).
- DEFINES - Keyword used to define new slot values for the event and default values.
- The slot names e_idsid_tool, e_method, e_level, and so on, are the slots defined in the event class.
- The default value is the slot facet that is used to define the type of information contained in the slot. The default value is used when the slot value is not assigned by the adapter. In this example, the slot names have default values assigned.

The following fields need to be set in the OR_LoginFailure.baroc file to achieve successful event correlation:

- **e_idsid_tool** This will be set to the name of the sensor type, which in this case we have classified as NT_Base. This entry is used to identify the different intrusion detection adapters.
- **e_method** Will always be set to 'knowledge'.
- **e_level** Represents a numerical value for the severity level. The default value is 1.0. This value is used in the weighting or importance determination relative to other events. For more information on this, please see Chapter 4, "Risk Manager correlation" on page 27.

- **severity** A TEC attribute used to set the event severity on the TEC and Risk Manager container entries. It has not been defined and will take on the default value of “HARMLESS” as defined in the riskmgr.baroc file.

6.4.10 Import new Oracle BAROC file into rulebase

The newly created OR_LoginfFailure BAROC file now must be imported into the active Risk Manager rulebase.

The following commands have to be executed in their respective order:

```
wimprbclass OR_LoginFailure.baroc <rulebase>
```

Importing the new or modified OR_LoginFailure.baroc into rulebase <rulebase>

```
wcomprules -t <rulebase>
```

will compile the BAROC files. The new files are now ready for use.

Now the TEC server must be restarted for the changes to take effect. This can be done either via the command line or the GUI interface on the Tivoli Enterprise Console.

The command line format is:

```
wstopesvr
sleep 10
wstartesrv
```

6.4.11 Test TEC with Oracle Adapter

The wtdumper command will list events send by the TEC NT adapter to the TEC event repository. The event for the OR_LoginFailure produces the following record in the event repository log after issuing the command:

```
wtdumper -o DESC
```

List received events in descending order.

```
0~0~ES~1~965744112(Aug 08 09:15:12 2000)~1~OR_LoginFailure~
LOGFILE~Oracle.ibm~9.3.187.215~tis002a~tis002a~N/A~OPEN~
~[ admin]~HARMLESS~
Aug 08 09:16:39 2000~
SOURCE: tis002a FAILED LOGIN ATTEMPT BY "TONROELANDSE", FROM 9.3.187.215, ACTION
 "100", RTNCODE 1017~
none~0~
```

Figure 70. OR_LoginFailure event repository reception

To see if the events are successfully submitted to TEC, check the events in the reception log with the wtdumprl command:

```
wtdumprl -o DESC
```

This will produce the following output:

```
1~9766~1~965744112(Aug 08 09:15:12 2000)
### EVENT ###
OR_LoginFailure;msg='SOURCE: tis002a FAILED LOGIN ATTEMPT BY "TONROELANDSE", FRO
M 9.3.187.215, ACTION "100", RTNCODE 1017';hostname=tis002a;origin=9.3.187.215;s
ub_source=Oracle.ibm;date='Aug 08 09:16:39 2000';sub_origin=tis002a;e_date='Aug
08 09:16:39';es_dst_haddress=tis002a;esr_src_haddress=9.3.187.215;e_idsid_hostna
me='"1"';esrsu_user='"TONROELANDSE"';e_signature='SOURCE: tis002a FAILED LOGIN A
TTEMPT BY "TONROELANDSE", FROM 9.3.187.215, ACTION "100", RTNCODE 1017';END

### END EVENT ###
PROCESSED
```

Figure 71. OR_LoginFailure reception log entry

The ‘PROCESSED’ statement indicates that the event has been successfully processed, and that the event will show up in the Risk Manager container as shown in Figure 72 on page 162. Detailed information on this event is shown in Figure 73 on page 163.

.

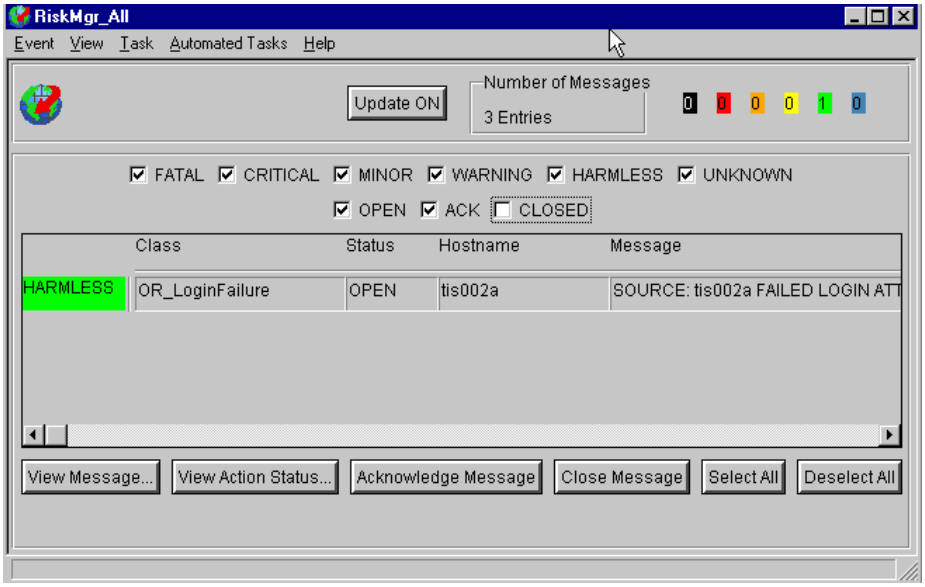


Figure 72. Risk Manager Event reception for OR_LoginFailure

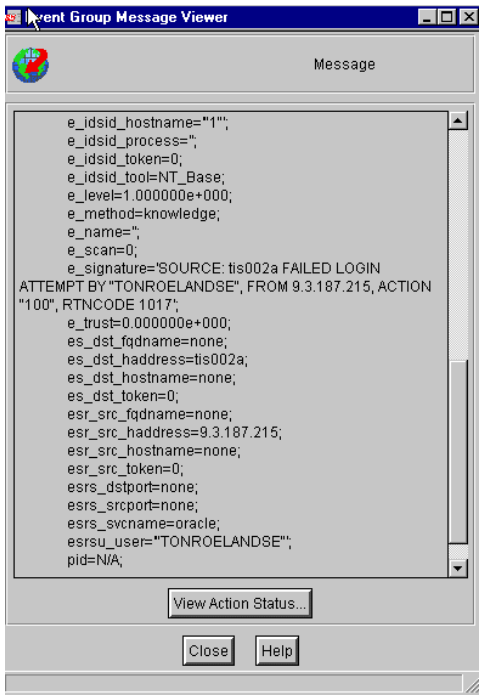


Figure 73. Risk Manager Event details for OR_LoginFailure

6.5 General comments on integration with Risk Manager

Some brief comments and tips on the integration of security related events into the Risk Manager correlation component.

- As with all kinds of information, investigate what you want to monitor, why you need it, and what to do with it.
- Security related events can come from a multitude of sources. The Risk Manager product comes with event adapters for Web IDS, ISS RealSecure, and Cisco IDS.
- Designing your own adapters is straightforward, but try to use the off the shelf adapters as much as possible.

6.5.1 What you should do

- DO derive as far down the idwg.baroc tree as is feasible. This gives the Risk Manager correlation component as much specific information as possible about the type of event.

- DO minimize the number of new attributes. Additional attributes may be created and may make sense for informational purposes, but the extra information will not be used by Risk Manager.
- DO derive only one level when deriving from the idwg.baroc tree. You can create a hierarchy of your own under the chosen ids_abstract class (and sometimes this may make sense), but, as mentioned above for additional attributes, Risk Manager will not use this additional information.
- DO make your class name identifiable with your adapter type. For example, all Oracle classes might begin with OR_ hence OR_LoginFailure.

6.5.2 e_date conversion

The e_date expects the time to be represented as the number of seconds from Jan 1, 1970. This is called the UNIX epoch time. In our test situation we did not have the utilities to do a conversion from event adapter time format which in our cases was a human readable date like Aug 3, 12:50:00 2000 instead of the more challenging 343556566.

It is our understanding that Tivoli Enterprise Console 3.7 has date conversion routines available. In the 3.6.2 we do not have these available and the way we solved this was by setting the e_date variable with the reception date of the event to TEC instead of the reception date by the adapter. This can be accomplished by the following rule:

```
rule: setd_rl:
(
   event: _evt of_class 'OR_LoginFailure'
          where [ date_reception: _date],
reception_action:
(
          bo_set_slot(_evt, e_date, _date),
          re_mark_as_modified(_evt,_),
          commit_rule
)
).
```

Figure 74. e_date rule setting

The brief explanation for this rule is that when OR_LoginFailure class is detected in the Event Repository, the date_reception will be pushed to the _date temporary variable. Then, on reception of the event in the reception log, the e_date variable will assume the value of _date (which contains the date_reception value), re-mark will make sure that this event will be updated on all TEC consoles in a dynamic fashion, and the commit statement will end this rule and continues with the next event or available rule.

There is a negative side to this approach besides additional load on the TEC server that has to do with the Risk Manager correlation component and the decay functionality.

When events come in gradually, they contain an event adapter time, which in the case of Risk Manager will be mapped to e_date, which holds the time the event occurred on the local host. When sent to TEC, the e_date does not change and therefore provides an accurate timestamp for correlation purposes. Both the correlation component and the decay function use this e_date field to determine situations (as explained in more detail in Chapter 4, “Risk Manager correlation” on page 27).

When using a rule to set the e_date equal to the TEC reception date, records received in batch will most certainly have the same date and time stamp, and this will affect the correlation accuracy and decay functionality.

Appendix A. Risk Manager class description and attribute list

This appendix lists the Risk Manager classes needed to create logfile adapters and to integrate the event feed of IDS and other security related events to the Risk Manager correlation component for processing. Some names might change over time, or other classes might have been added; at the time of creation of this book, specific information on these changes was not available for release.

Note that the prefix of the class name generally indicates its place in the class hierarchy. For example, for the class ESRM_ServiceRange the ESRM is derived from the E in IDS_Event, the S in E_SingleHost, the R in ES_RealOrigin and the M in ESR_MultipleServices. The base classes are listed at the end of each sub class series.

IDS_Event Classes derived from ids_abstract.baroc.
Please refer to Figure 52 on page 137 and Figure 53 on page 139 for a high level hierarchical overview of the attributes and relationships involved.

EM_RangeID

This class covers which network devices are involved in an alert. All network devices involved are identified by a HOSTID.

Attributes

emr_low emr_low_str	emr_high emr_high_str	emr_src emr_src_str

ESRM_ServiceRange

This class contains both lists of services that have been touched by the attack and lists of service ranges (for example intervals, most likely by service port, but also possibly by service name).

Attributes

esrms_low esrms_high esrms_svc

ESRSC_UserInfo

The suspicious command, which is what SC stands for, is associated with a user and password.

Attributes

esrscu_user esrscu_password

ESRSS_Activity

This class covers SNMP-related alerts when the intrusion detection sensor provides the entire PDU in addition to the community and object requested.

Attributes

esrssa_pdu

ESRST_UserInfo

In addition to the third host involved, the user triggering the alert is reported.

Attributes

esrstu_user esrstu_password

ESRSU_Passwd

A recognized password, this extends ESRS_UserDecode by adding an attribute for the password.

Attributes

esrsup_passwd

ESRSW_InsecureCgi

This class covers alerts reporting attempts that use cgi programs known to be vulnerable. For example, the requests for well known php and asp vulnerabilities are reported by alerts of this class.

Attributes

esrswi_cgi

ESRSW_PrivilegedCmd

This class covers alerts reporting attempted or successful shell or interpreter access through the web server (for example bat, sh, csh, perl).

Attributes

esrswp_command

ESRS_CmdDecode

This class covers alerts describing commands that are passed by users and are related to suspicious activity.

Attributes

esrsc_cmdline

ESRS_DistantCnt

This class covers alerts related to remote connections between two machines at the server level. An example of such traffic is NetBIOS.

Attributes

esrsd_client esrsd_server

ESRS_DnsCommand

This class covers DNS-related alerts. The number of DNS related alerts generated by intrusion-detection sensors has prompted the creation of this class.

Attributes

esrsd_domain

ESRS_EMail

This class covers e-mail related alerts. For example, this class covers alerts targeted at the sendmail or SMTP environments.

Attributes

esrse_address

ESRS_FileAccess

This class covers alerts related to file access

Attributes

esrsf_file

ESRS_PassDecode

This class covers alerts reporting passwords only when they cannot be attached to the associated account or user information directly. The service in this case is expected to be an authenticated service, such as telnet or ftp.

Attributes

esrsp_password

ESRS_RIP

This class covers routing related alerts. For more information see G. Malkin, "Rip Version 2," *Request for Comments (Standards track) 2453*, Internet Engineering Task Force, November 1998.

Attributes

esrsr_metric esrsr_route

ESRS_RemoteTools

Contains attributes for a local user and a remote user.

Attributes

esrsr_localuser esrsr_remoteuser

ESRS_SNMP

This class covers SNMP-related alerts. The number of SNMP-related alerts generated by intrusion-detection sensors has prompted the creation of this class. For more information on this topic see J. Case, R. Mundy, D. Partain, and B. Stewart, "Introduction to Version 3 of the Internet-Standard Network Management Framework," *Request for Comments (Informational) 2570*, Internet Engineering Task Force, April 1999.

Attributes

esrss_community esrss_oid

ESRS_StringMatch

This class covers alerts produced by an intrusion detection sensor matching a given pattern with an input stream (described in the service). One example of this is the decoding of text protocols by certain intrusion detection sensors, which then generate an alert when a given string in the flow matches.

Attributes

esrss_content

ESRS_ThirdHost

Certain alerts report attacks that involve three hosts. An example of such attack is ftp bounce. This class reports the identity of the machine used in the bounce.

Attributes

esrst_host esrst_service

ESRS_Trojan

This class covers trojan detection alerts. Alerts generated involve detection of Back Orifice, NetBus, and other well-known trojans.

Attributes

esrst_trojan esrst_command esrst_args

ESRS_UserDecode

This class covers user-related alerts. Examples of such alerts include logins (telnet, ftp, login, ssh) or r- services.

Attributes

esrsu_user

ESRS_WebServer

This class and its subclasses group activity related to Web server attacks. This includes all Web-related traffic, and most of the html documentation servers in UNIX environments running on port 8888 or port 8080. There are many alerts generated by this service, so it has been specialized in the class hierarchy.

Attributes

esrsw_url

ESR_ICMP

This class groups attacks related to ICMP traffic. The typical attacks covered by this class are ping floods (or other kinds of ICMP floods), but also BGP, EGP, and other low level protocols. ARP/RARP is also a candidate here.

Attributes

esri_code esri_type

ESR_IP

This class deals with unknown IP activity.

Attributes

esri_protocol esri_reason

ESR_MultipleServices

This class covers attacks that involve two machines, but span over a list of services or applications. The total number of services or applications involved must be reported.

Attributes

esrm_number

ESR_SingleService

This class covers all intrusion-detection alerts that contain the triplet (destination, source, service). As such, most intrusion detection alerts will be mapped onto subclasses of this class.

Attributes

esrs_srcport esrs_dstport esrs_svcname

ESR_Tool

This class groups information concerning the detection of tool activity, such as SATAN, ISS, or others.

Attributes

esrt_tool

ESSM_ServiceRange

When a range is specified, the number field associated with the parent class must contain the size of the range, for example the number of services comprised between the low and high services. When ranges are specified, they should indicate that all services in the range have been affected by the alert.

Attributes

essms_low essms_high essms_svc	essms_low_str essms_high_str essms_svc_str

ESS_MultipleServices

This class covers alerts indicating that multiple services have been attacked from a spoofed address. The minimal information to report there is the number of services that have been probed or involved. The lists and ranges of elements that have been probed or involved are reported in a subclass.

Attributes

essm_number

ESS_SingleService

This class describes a service being attacked from a spoofed address

Attributes

esss_srcport esss_dstport esss_svcname

ES_Application

This class represents alerts that are happening on the local machine. In most cases, this means that the attack is being run locally. Examples of such alerts include reading passwords on a Windows 95 machine, the SUN loadmodule, and certain symlink vulnerabilities. An alert of this class means that the attack/anomaly is carried out locally. It does not mean that the attacker/perpetrator is local to the device. For example, the loadmodule attack would be reported by the same alert, irrespective of whether the user is logged in on the console (physically present) or remotely connected via telnet.

To report the second case completely, two such alerts have to be generated; one for the connection and one for the local action. Most intrusion-detection systems should provide alerts in this part of the class hierarchy, which is the most detailed.

Attributes

esa_appli

ES_RealOrigin

This class contains alerts for which there is no way to differentiate between a spoofed origin and the real source of the alert. This does not mean that the intrusion detection sensor makes any guaranty that the source given in the alert is the one that actually carried out the alert.

Attributes

esr_src_hostname esr_src_fqdname esr_src_haddress esr_src_token

ES_SpoofedOrigin

From the nature of the signature, the source host information is known to have been falsified (spoofing). A typical example of this class of alerts is the Land attack, when source and destination addresses are the same. When it is not possible to determine if the address is false, then the AS_REALORIGIN subclass hierarchy is used (even though the address may still be wrong).

Attributes

ess_src_hostname ess_src_fqdname ess_src_haddress ess_src_token

E_Weird (will be replaced by E_Internal)

This class is provided for reporting internal IDS sensor anomalies. This class could be used for reporting IDS sensor exceptions or events that might represent an attack against the IDS sensor.

Attributes

Same as IDS_Event

E_MultipleHosts

Multiple host destinations (or targets) are involved. An example would be a wide port scan using TCP or UDP.

This class is intended to group more complex alerts that concern several targets in a single domain. An example of such an alert would be a system wide scan where one source tries to connect to the same service on several pieces of target equipment. Destination identifications are provided by sub classing

Attributes

em_number

E_SingleHost

A single host destination (or target) is involved.

Most simple alerts coming from intrusion detection sensors will be in this hierarchy. The first component is the destination/target of the alert. The destination of the attack is either the recipient of the action in the case of a

remote attack (for example destination of the IP packet, host name to which the attacker logs in), or the host on which the action is carried out in the case of a local attack. Remote attacks for which the intrusion detection system has no information concerning the remote host should be reported at this level or using the AS_APPLICATION subclass hierarchy.

Attributes

es_dst_hostname
es_dst_fqdname
es_dst_haddress
es_dst_token

IDS_Event

Top level class from the IETF Intrusion Detection Working Group (IDWG) hierarchy draft.

The alert class defines the bare minimum information that every intrusion-detection system must provide for every alert to be IDWG-compliant. However, intrusion-detection systems should use subclasses to report alert information. Sub classing or using the alert class is only acceptable in two cases; when the intrusion detection system wishes to report some internal condition that has no destination or source involved (for example out of memory, out of disk, dump core, etc.), or when an entirely new class of intrusion-detection systems appear. Because it is extremely likely that alerts should contain a target, the later case should not happen before the standard is revised.

The alert class does not carry information about the success of the alert, i.e. if the compromise attempt was successful or not. The rational for not including this information is that it is not relevant to all alerts. However, alerts can indicate a measure of success by a combination of severity and priority.

Attributes

e_flood	e_idsid_process
e_date	e_idsid_token
e_date_int	e_signature
e_idsid_tool	e_name
e_cons	e_level
e_scan	e_trust
e_idsid_hostname	e_method
e_idsid_haddress	

IDS_Generic Classes (ids_generic.baroc)

IDS_Generic

A general purpose class for reporting an event, derived from IDS_Event (an abstract class). It could be used as a base class for generic events related to network-based ids, host-based ids, routers, firewalls, etc. It is designed to facilitate rapid development of an adapter. An adapter developer could use this class initially before developing a BAROC class file specifically for the sensor.

Note

This would require that e_idsid_tool (or sub_source, from which e_idsid_tool could be derived) be set before the event reaches the server.

Correlation Classes (riskmgr.baroc)

A_Situation

Base class for situation events.

Attributes

ag_idsids ag_sigs	ag_date ag_level	ag_decay ag_type

A_Situation1

Categorized by signature/source/destination.

Attributes

ag_token1 ag_string1	ag_token2 ag_string2	ag_token3 ag_string3

A_Situation2

Categorized by destination/source, signature/source, or signature/destination.

Attributes

ag_token1 ag_string1	ag_token2 ag_string2	ag_token3s ag_string3s

A_Situation3

Categorized by signature, source, or destination.

Attributes

ag_token1 ag_string1	ag_token2s ag_string2s	ag_token3s ag_string3s

A_TrustedHost

Activity detected from a host that is considered trusted. Does not raise an alert for activity from this host, but maintains a list of destination hosts related to this source host.

Aggregate

Base class for classes that aggregate information (idsid list, signature list), A_Situation, and A_TrustedHost.

Attributes

ag_idsids ag_sigs

C_Idsid

Identifier for intrusion detection sensor instances.

Attributes

idsid_tool idsid_hostname	idsid_haddress idsid_proces

ComplexType

Base class for C_Idsid.

Attributes

ct_token ct_counter

IDS_Base

Base class for IDS_Event. Used to set default values for various attributes of the class EVENT.

Also used to extend the EVENT class by adding several attributes that are used for correlation and should be considered "hidden" or "protected" from subclasses.

Attributes

server_handle	date	num_actions
date_reception	status	repeat_count
event_handle	administrator	cause_date_reception
source	acl	cause_event_handle
sub_source	credibility	pid
origin	severity	
sub_origin	msg	
hostname	msg_catalog	
adapter_host	msg_index	
date	duration	

IE_BadInput

This class captures errors on input data.

Attributes

repeat_count	error_file
cause_date_reception	error_line
cause_event_handle	

IE_PrologFailure

This class captures Prolog failures

Attributes

error_file
error_line

IE_Situation

An event of this class is generated when the situation processing fails because of an unknown reason. The message of the event indicates in which part of the processing stage the failure occurred.

Attributes

error_file error_line situation_kind situation_type token1	token1_string token2 token2_string token3 token3_string	token_ids token_ids_string situation_msg event_signature

IE_TokenOverflow

This class captures overflowing tokens.

Attributes

repeat_count cause_date_reception cause_event_handle	error_file error_line

InternalError

Base class for reporting errors.

Attributes

repeat_count cause_date_reception cause_event_handle	error_file error_line

A.1 Revisions and pre-release notes next version

When developing this redbook and working with Risk Manager Version 1.0, the Austin development team initiated changes in the class hierarchy that improve the structure and flexibility of the Risk Manager product. Unfortunately, we were not able to test this code and benefit from these changes, but the redbook team and the Tivoli Risk Manager development team decided to share this information and to provide more insight with regards to adapter changes.

By the time this redbook gets published, the changes in the hierarchy will be included in the new release of Risk Manager.

Note

We have tried to include the most significant changes in the Risk Manager hierarchy but we cannot provide any guarantee on the accuracy of these changes.

A.1.1 Improved Hierarchy structure

The top level hierarchy is shown in Figure 75.

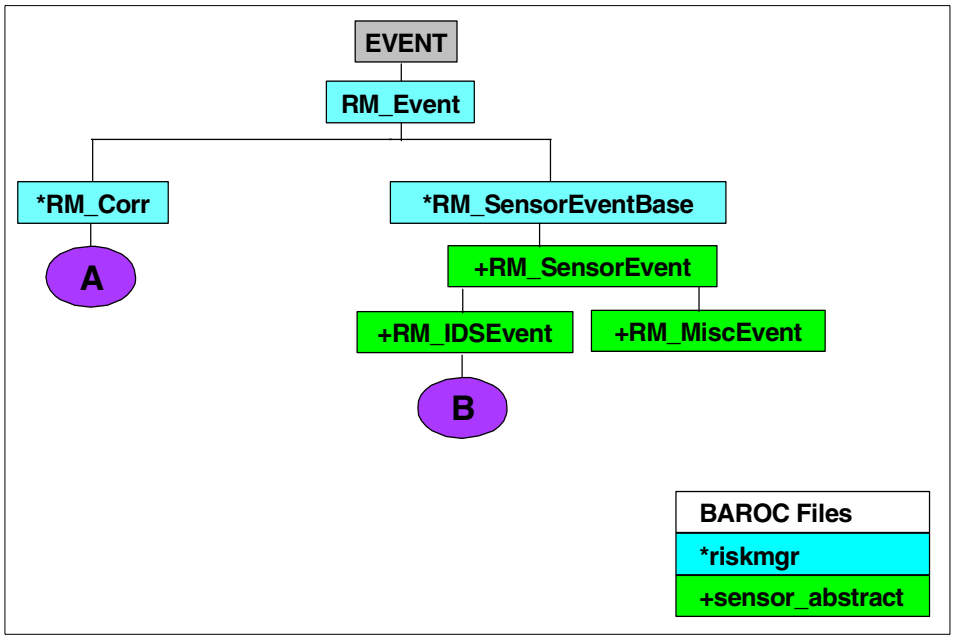


Figure 75. Top Risk Manager attribute hierarchy (new release)

As shown in Figure 75, RM_Event has been added as the Risk Manager top level class, and the prefix has changed from E to RM_ across the board.

The following names have been changed:

Old Name	New Name
-	RM_Event
IDS_Base	RM_SensorEventBase
Aggregate	RM_Corr

Old Name	New Name
-	RM_SensorEvent
IDS_Event	RM_IDSEvent
-	RM_MiscEvent

RM_Event

Top Level Class for all Risk Manager based events. This class inherits default TEC attributes from the EVENT super class.

Attributes

RM_Version	RM_Timestamp

RM_SensorEventBase

Base class for sensor related events. Defines new attributes used internally for correlation.

Attributes

rm_Timestamp32 rm_SourceToken	rm_SensorToken: rm_ConsequenceList	rm_DestinationToken

RM_Corr

Base class for correlation related events such as situations, trusted hosts, and so on. This event is related to RM_Event.

RM_SensorEvent

Base class for sensor related events.

Attributes

rm_SensorType rm_SensorHostname rm_SensorIPAddr	rm_SensorPID rm_SensorOS rm_Signature	rm_Description rm_Level rm_Correlate

RM_IDSEvent

Base class for IDS related events.

Attributes

rm_NameData rm_SourceHostname rm_SpoofedSource rm_NameType	rm_DestinationHostname rm_SourceIPAddr rm_NameID rm_DestinationIPAddr

RM_MiscEvent

Base Class describing miscellaneous events.

Attributes

rm_Correlate rm_Category	rm_ObjectType rm_Object	rm_Action

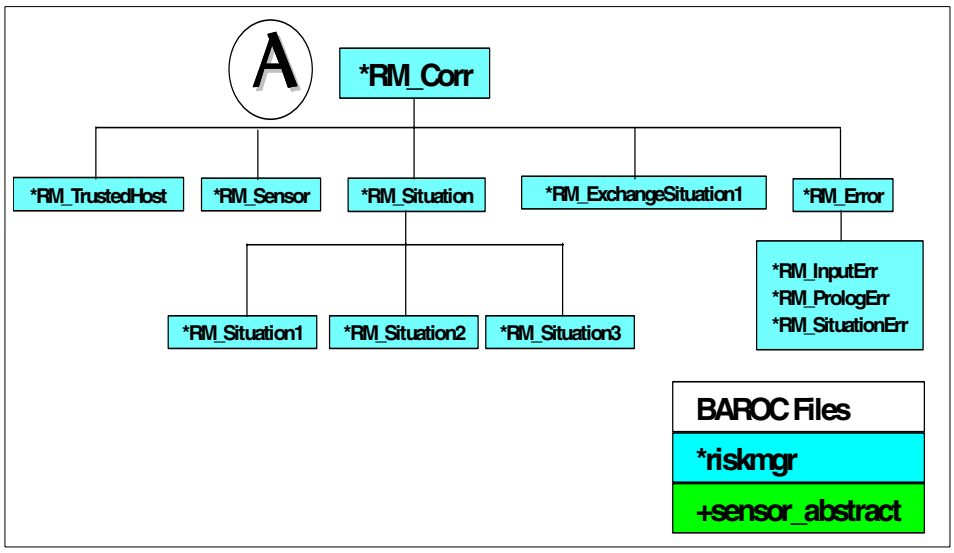


Figure 76. RM_Corr attribute hierarchy (new release)

The following names have been changed:

Old Name	New Name
Aggregate	RM_corr
A_TrustedHost	RM_TrustedHost
C_IDSID	RM_Sensor
A_Situation	RM_Situation
-	RM_Exchange_Situation1
-	RM_Error
A_Situation1	RM_Situation1
A_Situation2	RM_Situation2
A_Situation3	RM_Situation3
IE_BadInput	RM_InputErr
IE_PrologFailure	RM_PrologErr
IE_Situation	RM_SituationErr

RM_Corr

Base class for correlation related events such as situations, trusted hosts, and so on. This event is related to RM_Event.

RM_TrustedHost

Class representing a trusted host.

Attributes

rm_HostToken rm_Hostname rm_HostIPAddr rm_SignatureList	rm_DestinationTokenList rm_DestinationHostnameList rm_SensorList rm_DestinationHostIPAddrList

RM_Sensor

Class representing a sensor. The sensor instance information is also kept in prolog facts. This class is to allow administrators to see sensors at the console and for later database searching. rm_SensorNew='no' and severity='HARMLESS' indicates that the sensor was expected (i.e. defined in riskmgr_hosts.pro) or Risk Manager was configured to ignore the automatic

discovery of the sensor instance. For unexpected sensor instances, rm_SensorNew should be 'yes' and the severity should be 'MINOR'.

Attributes

RM_SensorNew

RM_Situation

Base class for situations. Situation events are the result of the aggregation and correlation process.The aggregation and correlation process can also result in the modification of existing situation events.

Attributes

rm_SensorList rm_SignatureList rm_Key1	rm_Level rm_Decay rm_Key1Str	rm_Timestamp32

RM_ExchangeSituation1

Class used for events that send situation1 events to another TEC server.

Attributes

rm_ClassToken rm_TargetToken rm_TargetHostname m_TargetIPAddr	rm_SourceToken rm_SourceHostname rm_SourceIPAddr m_SensorTokenList	rm_TimeUpdated rm_SignatureList rm_DecayList

RM_Error

Base class for error events

Attributes

rm_ErrFile	rm_ErrLine	rm_ErrMethod

RM_Situation1

Class representing a Situation1 event.

Attributes

rm_Type rm_Key1	rm_Key2 rm_Key2Str	rm_Key3 rm_Key3Str

RM_Situation2

Class representing a Situation2 event.

Attributes

rm_Type rm_Key2	rm_Key2Str rm_Key3List	rm_Key3ListStr

RM_Situation3

Class representing a Situation3 event.

Attributes

rm_Type rm_Key2List	rm_Key2ListStr rm_Key3List	rm_Key3ListStr

RM_InputErr

Class representing input errors. It is derived from RM_Error, and is similar in structure with more detailed information about the error in the EVENT msg attribute.

RM_PrologErr

Class representing Prolog errors. It is derived from RM_Error, and is similar in structure with more detailed information about the error in the EVENT msg attribute.

RM_SituationErr

Class representing situation related errors

Attributes

rm_SensorTokenStr	rm_SituationName	rm_SituationType

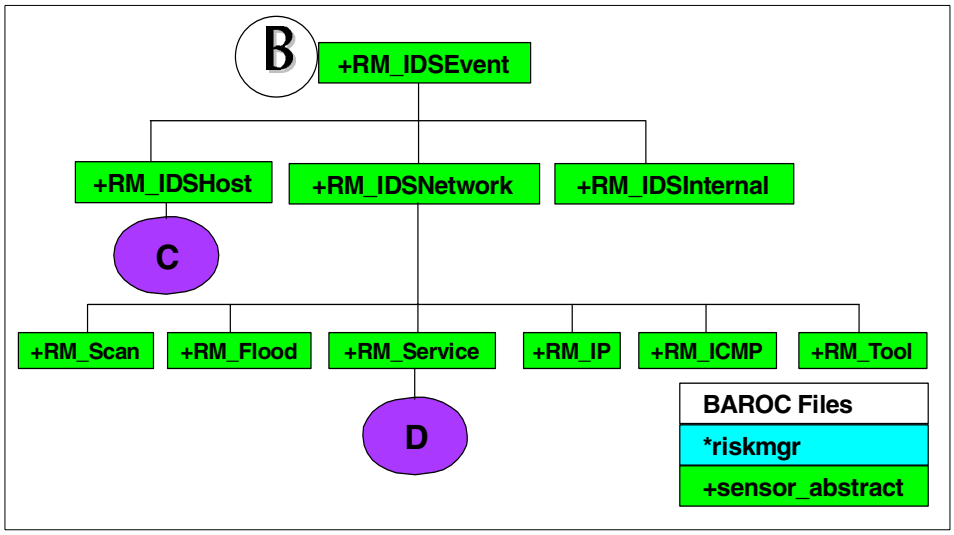


Figure 77. RM_IDSEvent hierarchy (new release)

The following names have been changed:

Old Name	New Name
IDS_Event	RM_IDSEvent
E_SingleHost	RM_IDSHost
E_MultipleHost	RM_IDSNetwork
E_Internal	RM_IDSInternal
-	RM_Scan
-	RM_Flood
ESR_SingleService	RM_Service
ESR_IP	RM_IP
ESR_ICMP	RM_ICMP
ESR_Tool	RM_Tool

RM_IDSEvent

Top level class from the IETF Intrusion Detection Working Group (IDWG) hierarchy draft.

The alert class defines the bare minimum information that every intrusion detection system must provide for every alert to be IDWG compliant. However, intrusion-detection systems should use subclasses to report alert information. Sub classing or using the alert class is only acceptable in two cases; when the intrusion detection system wishes to report some internal condition that has no destination or source involved (for example out of memory, out of disk, dump core, etc.), or when an entirely new class of intrusion-detection systems appear. Because it is extremely likely that alerts would contain a target, the later case should not happen before the standard is revised.

The alert class does not carry information about the success of the alert, i.e. if the compromise attempt was successful or not. The rational for not including this information is that it is not relevant to all alerts. However, alerts can indicate a measure of success by a combination of Severity and Priority.

Attributes

rm_Level rm_Correlate m_NameType rm_NameID rm_NameData	rm_DestinationHostname rm_DestinationIPAddr rm_SourceHostname rm_SourceIPAddr m_SpoofedSource

RM_IDSNetwork

Base class for network type activity (i.e., when both a source and a destination host are involved). Either network-based or host-based IDS products may report this type of information. The majority of events classified here are expected to be from network-based IDS products.

Attributes

RM_Protocol

RM_IDSInternal

Class provided for reporting internal IDS sensor anomalies.

Attributes

rm_Description

RM_Scan

Class representing a scan.

Attributes

rm_PortCount

RM_Flood

Class representing a flood. It inherits all attributes from RM_IDSNetwork.

RM_Service

Description not available.

Attributes

rm_Protocol rm_DstPort	rm_SrcPort rm_Servicename

RM_IP

This class deals with unknown IP activity.

Attributes

rm_Protocol	Reason

RM_ICMP

This indicates ICMP type attacks, such as ping floods.

Attributes

rm_ICMPCode	rm_ICMPType

RM_Tool

Class that holds tool names.

Attributes

rm_Toolname

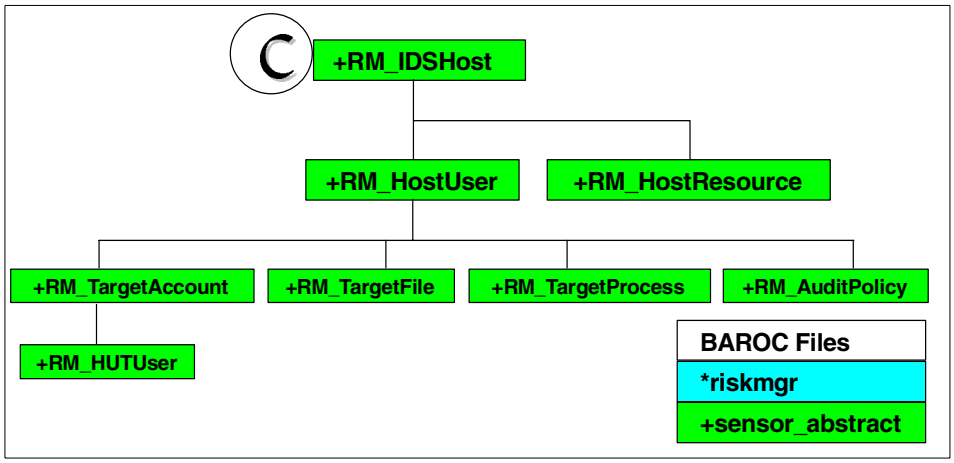


Figure 78. RM_IDSHost hierarchy (new release)

The following names have been changed:

Old Name	New Name
ES_Application	RM_IDSHost
-	RM_HostUser
-	RM_HostResource
-	RM_TargetAccount
-	RM_TargetFile
-	RM_TargetProcess
-	RM_AuditPolicy
-	RM_HUTUser

RM_IDSHost

Most simple alerts coming from intrusion detection sensors are expected to be in this hierarchy. The first component being highlighted in the alert is the destination/target of the alert. The destination of the attack is either the recipient of the action in the case of a remote attack (for example destination of the IP packet, host name to which the attacker logs in), or the host on which the action is carried out in the case of a local attack.

Attributes

rm_PtyInfo rm_SrcPort	rm_DstPort rm_Servicename	rm_PID

RM_HostUser

Attributes

rm_HUsername rm_HUserID	rm_HUserDomain rm_HUPurpose	rm_HUAdditional

RM_HostResource

Attributes

rm_Name	rm_State

RM_TargetAccount

Attributes

rm_HUTAccountname rm_HUTPurpose rm_HUTAccountDomain	rm_HUTAccountID rm_HUTAdditional

RM_TargetFile

Attributes

rm_HUTFilename	rm_HUTAccessFlags

RM_TargetProcess

Attributes

rm_HUTProcessID	rm_HUTProcessname

RM_AuditPolicy

Class representing audit policy events.

Attributes

rm_SystemSuccess rm_SystemFailure rm_LogonSuccess rm_LogonFailure rm_ObjectAccessS	rm_ObjectAccessF rm_PrivilegeUseF rm_PrivilegeUseS rm_DetailedTrackingS rm_DetailedTrackingF	rm_PolicyChangeS rm_PolicyChangeF rm_AccountMgmtS rm_AccountMgmtF

RM_HUTUser

Attributes

rm_HUTUAccountname	rm_HUTUAccountDomain	rm_HUTUAdditional
rm_HUTUAccountID	rm_HUTUPurpose	

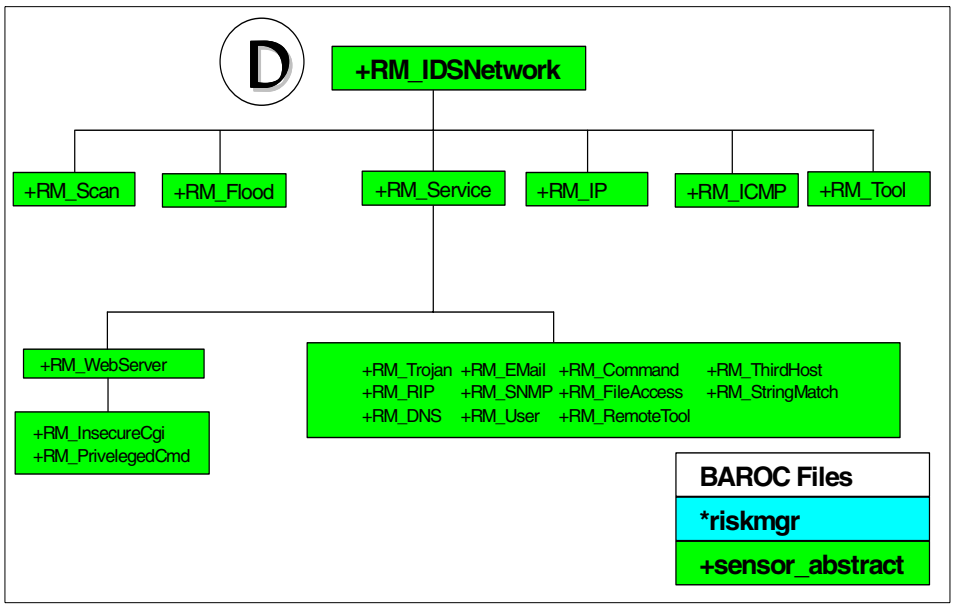


Figure 79. RM_IDSnetwork hierarchy new release

The following names have been changed:

Old Name	New Name
E_MultipleHost	RM_IDSNetwork
-	RM_Scan
-	RM_Flood
ESR_SingleService	RM_Service
ESR_IP	RM_IP
ESR_ICMP	RM_ICMP
ESR_Tool	RM_Tool
ESRS_WebServer	RM_WebServer
ESRSW_InsecureCgi	RM_InsecureCgi
ESRSW_PrivilegedCmd	RM_PrivilegedCmd
ESRS_Trojan	RM_Trojan
ESRS_RIP	RM_RIP
ESRS_DnsCommand	RM_DNS
ESRS_EMail	RM_Email
ESRS_SNMP	RM_SNMP
ESRS_UserDecode	RM_User
ESRS_CmdDecode	RM_Command
ESRS_FileAccess	RM_FileAccess
ESRS_RemoteTools	RM_RemoteTool
ESRS_ThirdHost	RM_ThirdHost
ESRS_StringMatch	RM_StringMatch

RM_IDSNetwork

Base class for network type activity (i.e., when both a source and a destination host are involved). Either network-based or host-based IDS products may report this type of information.

Attributes

rm_Protocol

RM_Scan

Class representing a scan.

Attributes

rm_PortCount

RM_Flood

Class representing a flood.

Attributes

rm_Flood

RM_Service

This class covers all intrusion-detection alerts that contain the triplet (destination, source, service). As such, most intrusion detection alerts will be mapped onto subclasses of this class.

Attributes

rm_Protocol rm_DestPort	rm_SrcPort rm_servicename

RM_IP

This class deals with unknown IP activity.

Attributes

rm_Protocol	rm_Reason

RM_ICMP

This indicates ICMP type attacks, such as ping floods.

Attributes

rm_ICMPCode	rm_ICMPType

RM_Tool

This class groups information concerning the detection of tool activity, such as SATAN, ISS, or others.

Attributes

rm_Toolname

RM_InsecureCgi

The CGI Script name.

Attributes

rm_Cgi

RM_PrivelegedCmd

The command being tried (Web server related).

Attributes

rm_Command

RM_Trojan

This class covers trojan detection alerts such as detection of Back Orifice, NetBus, and other well-known trojans.

Attributes

rm_Trojan	rm_command	rm_Args

RM_RIP

This class covers routing related alerts. For more information, see G. Malkin, "Rip Version 2," *Request for Comments (Standards track) 2453*, Internet Engineering Task Force, November 1998.

Attributes

rm_Metric	rm_Route

RM_DNS

The DNS domain name.

Attributes

rm_Domain

RM_User

This class covers user-related alerts. Examples of such alerts include logins (telnet, ftp, login, ssh) or r- services.

Attributes

rm_User	rm_Password

RM_Command

This class covers alerts describing commands that are passed by users and are related to suspicious activity.

Attributes

rm_Command

RM_FileAccess

This class covers alerts related to file access.

Attributes

rm_File

RM_ThirdHost

Certain alerts report attacks that involve three hosts. An example of such attack is ftp bounce. This class reports the identity of the machine used in the bounce.

Attributes

rm_ThirdHost	rm_ThirdPort

RM_StringMatch

This class covers alerts produced by an intrusion detection sensor matching a given pattern with an input stream (described in the service). One example of this is the decoding of text protocols by certain intrusion detection sensors, which then generate an alert when a given string in the flow matches.

Attributes

rm_Content

Appendix B. Common Vulnerabilities and Exposures (CVE)

CVE is a joint effort of the IT security community to provide a list of common vulnerabilities and exposures. Its founders refer to it as an “aid for coordination of protection in cyberspace”. It was founded in 1999, and addresses an issue that has troubled this community for a long time.

There is a multitude of intrusion detection tools, vulnerability databases and security incident reporting services. Each of these uses different names for their subjects, even though they may be referring to the same situation. As a result, it is difficult to match and correlate these different sources of information. Duplicate work, lack of reliable information, and poor communication between the experts is the consequence.

Note

CVE provides a common dictionary for vulnerability and exposure names, called the *CVE list*. It does not provide any technical content; it merely serves as an index to enhance the usability of the content that already exists. It provides references to sources of information where technical details, findings, discussions, solutions, bugfixes, and various other aspects regarding known vulnerabilities and exposures can be found.

The CVE Web site holds information about CVE itself, the CVE list, and a wealth of links to related information and activities. It is located at `www.cve.mitre.org` on the World Wide Web.

B.1 Organizational structure of CVE

CVE was launched by an initiative headed by MITRE. MITRE is a US-based not-for-profit corporation working in partnership with government clients.

MITRE created the Editorial Board, maintains CVE with assistance from the Editorial Board, and provides neutral guidance throughout the process.

The CVE Editorial Board, chaired by a MITRE representative, includes representatives from over 20 security-related organizations such as security tool vendors, incident response teams, academic institutions, intrusion detection experts, and security analysts. The Board meets approximately once per quarter. The list of members can be found at the CVE web site.

The CVE Editorial Board determines when a vulnerability or exposure should have an entry in CVE. This is based on publicly available information. Members discover security problems in the course of their own work; from reading discussion lists, technical documents, and vendor alerts; and word of mouth. Any member can nominate a candidate to be added to CVE.

The Editorial Board determines when a problem is truly unique and well understood. Based on this principle a voting decision is taken to add it to the list. Because CVE is designed to contain mature information, it relies on other mechanisms to handle newly emerging information. New entries will make it into CVE once they have been verified through these other mechanisms. Examples of such mechanisms are vulnerability analysis teams, an emergency response team such as CERT, or other organizations that are designed to handle new information. Many of these organizations have a presence on the Editorial Board.

B.2 Terminology

CVE distinguishes between two categories based on the observation that the word “vulnerability” is being used in a different context by different people:

- *Universal vulnerabilities* are those problems that are considered a vulnerability under *any* commonly used security policy that includes at least some requirements for minimizing the threat from an attacker.
- *Exposures* in contrast are problems that are *sometimes* thought of as vulnerabilities in *some* security policies.

This distinction has not yet been defined in a strict way, but there are guidelines that can be applied to any specific case. Under these a universal vulnerability would allow an attacker to:

- Execute commands as another user
- Access data that is contrary to the specified access restrictions for that data
- Pose as another entity
- Conduct a denial of service.

An exposure is a state in a computing system (or set of systems) that is not a universal vulnerability, but:

- Allows an attacker to conduct information gathering activities
- Allows an attacker to hide activities

- Includes a capability that behaves as expected, but can be easily compromised
- Is a primary point of entry that an attacker may attempt to use to gain access to the system or data
- Is considered a problem according to some reasonable security policy

Examples

Examples of universal vulnerabilities provided by CVE include:

- phf (remote command execution as user "nobody")
- rpc.ttdbserverd (remote command execution as root)
- World-writable password file (modification of system-critical data)
- Default password (remote command execution or other access)
- Smurf (Denial of Service by flooding a network)

Examples of exposures include:

- Running services such as finger (useful to an attacker for information gathering, though it works as designed)
- Running services that are common attack points (e.g. HTTP, FTP, or SMTP)
- Use of applications or services that can be successfully attacked by brute force methods (for example, weak passwords)

Note

At present, CVE provides no mechanism to distinguish between vulnerabilities and exposures. This may change in the future. Users of the CVE list are encouraged to distinguish between CVE entries in any manner that best supports their own particular objectives.

B.3 CVE List

CVE is a list of information security vulnerabilities and exposures that aims to provide common names for publicly known problems. The goal of CVE is to make it easier to share data across separate vulnerability databases and

security tools with this "common enumeration." The CVE list is a table of three columns. It looks like this:

Table 17. The CVE list (sample entries)

Name	Description	References
CVE-1999-0002	Buffer overflow in NFS mountd gives root access to remote attackers, mostly in Linux systems.	SGI:19981006-01-I CERT:CA-98.12.mountd CIAC:J-006 BID:121 XF:linux-mountd-bo
CVE-1999-0003	Execute commands as root via buffer overflow in Tooltalk database server (rpc.ttdbserverd).	NAI:NAI-29 CERT:CA-98.11.tooltalk SGI:19981101-01-A SGI:19981101-01-PX XF:aix-ttdbserver XF:tooltalk BID:122
CVE-1999-0067	CGI phf program allows remote command execution through shell metacharacters.	CERT:CA-96.06.cgi_example_code XF:http-cgi-phf BID:629
CVE-1999-0751	Buffer overflow in Accept command in Netscape Enterprise Server 3.6 with the SSL Handshake Patch.	BUGTRAQ:19990913 Accept overflow on Netscape Enterprise Server 3.6 SP2 BID:631
CVE-2000-0414	Vulnerability in shutdown command for HP-UX 11.X and 10.X allows local users to gain privileges via malformed input variables.	HP:HPSBUX0005-113 XF:hp-shutdown-privileges BID:1214

The first column provides the CVE name. The first part of the name is a constant (CVE), the second part indicates the year of creation, the third part is a number within that year.

The second column gives a short verbal description to aid searching of the CVE list.

The third column contains a list of references, starting with a short name for the information source, followed by a colon, and concluding with the identifier according to the numbering or naming scheme used by the respective information source.

In addition to the CVE list there is the CVE candidate list. It has the same structure, only the entry names have CAN as their constant as opposed to CVE. These entries are still under evaluation. If they make it to the final list, they will be given a different name.

The CVE Web site provides a search engine for keyword search of both CVE and candidate lists at the same time. This is one method of searching. It has the advantage that search arguments can be combined. Alternatively, the list can be downloaded and used offline.

B.3.1 How to use the CVE list

Take, for example, the vulnerability listed as "CVE-1999-0067”. Various manufacturers and organizations refer to this single vulnerability as "http-cgi-phf", "http_escshellcmd", "CERT: CA-96.06.cgi_example_code", "#3200-WWW phf attack", or "Vulnerability in NCSA/Apache Example Code." as shown in Figure 80. Using one accepted reference (the CVE number) will obviously make it easier to communicate and correlate security information. You can search CVE for specific vulnerabilities and exposures. To evaluate products, you can download CVE, identify CVE entries related to your evaluation needs, and directly compare CVE-compatible tools and databases.

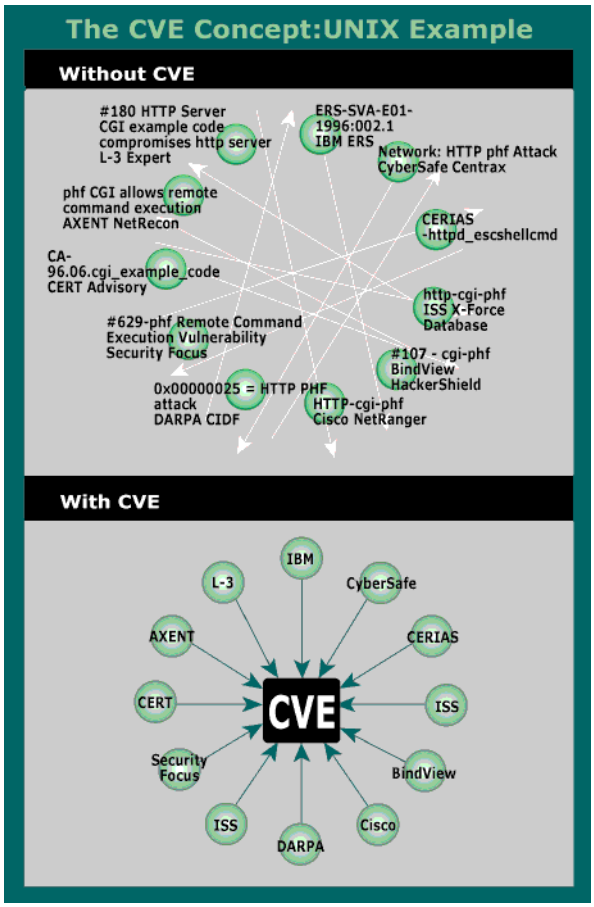


Figure 80. The CVE concept (Source: www.mitre.org)

The CVE Web site facilitates the exploitation of the joint effort of the CVE contributors by providing a list of compatible products.

B.3.2 Useful web links

During the residency for this redbook we used the following Web sites to work with CDE related information:

- Sites that were searchable with CVE names and provided useful information as well as direct links to other providers' records:

  `csrc.nist.gov/icat/`
  `www.securityfocus.com`

- Sites that directly relate to the short name in the reference entries of the CVE list. The identifier following the short name can be used as a search word on the corresponding web site. We only provide a few examples. Actual URLs may change any time.

BID	`securityfocus.com/bid/<nnn>` (replace <nnn> by actual number)
CERT	`www.cert.org/advisories/`
XF	`xforce.iss.net`
AIXAPAR	See Section B.3.2.1, “Finding an APAR on the IBM web site” on page 205
IBM	See Section B.3.2.2, “IBM Emergency Response Service (ERS)” on page 206
MS and **MSKB**	`www.microsoft.com/technet/security/current.asp`
RSI	`www.codetalker.com/advisories/rsi`
SUN	`sunsolve.sun.com`
AUSCERT	`www.auscert.org.au`
CIAC	`ciac.llnl.gov`
SGI	`www.sgi.com/support/security/advisories.html`

B.3.2.1 Finding an APAR on the IBM web site

We used the search engine available on the IBM Web site to locate the APAR (Authorized Program Analysis Report). This method has the advantage that it does not rely on the current structure of the Web site, which may change in the future. We started at `www.software.ibm.com`, selected **Support** from the horizontal menu at the top. On the Support page we selected the link named **Advanced search** which brought us to the Advanced Search page. We used the APAR number (in our example IX80543) as the search word, selected document type **APAR (Authorized Program Analysis Report)** and product group **Software**.

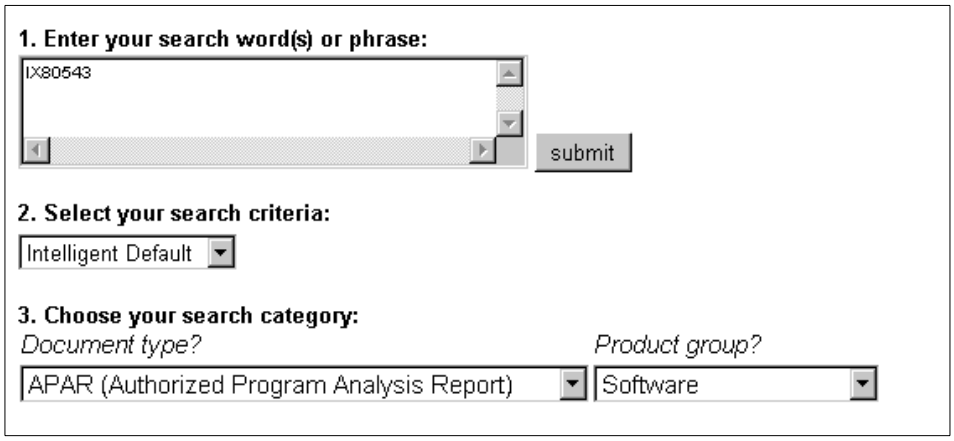


Figure 81. APAR search on IBM Software web site

B.3.2.2 IBM Emergency Response Service (ERS)

IBM Global Services has a group of service offerings in place called Emergency Response Services. One part of this initiative is the creation and distribution of ERS advisories. These are security and vulnerability advisories and other security-related information that are distributed to customers. These advisories are designed to support a proactive security policy. They may be accessed and downloaded by the general public, although redistribution restrictions are contained in the respective documents.

The CVE list entries that use IBM as a reference identifier refer to this resource. The URL for the advisories is `www-1.ibm.com/services/continuity/recover1.nsf/advisories`.

B.3.3 CVE-compatible products

By products CVE means tools, Web sites, and databases. CVE-compatible is defined here in the sense that a product uses CVE names in a way that allows it to cross-link with other repositories that use CVE names. To be CVE-compatible, a product must meet *all* of the following requirements:

1. *CVE Searchable*: A user can search using a CVE name to find related information.
2. *CVE Output*: Information is presented that includes the related CVE name(s).
3. *Mapping*: The repository owner has provided a mapping relative to a specific version of CVE, and has made a good faith effort to ensure accuracy of that mapping.

The CVE Web site provides a list of organizations that have declared that they are working to make their product/database CVE-compatible. For each member on the list details are given regarding that declaration.

Note

It is Tivoli's stated intention to make Risk Manager CVE-compatible. Here is the wording of the declaration:

Tivoli actively promotes, supports, and contributes to the emerging open systems standards such as CVE that enable technology management software such as Tivoli SecureWay Risk Manager, intrusion detection, and vulnerability assessment components to inter-operate and share management information. Tivoli SecureWay Risk Manager uses an IDEF-compliant data-format for communication with ID sensors. The IDEF CLASSIFICATION Class names the vulnerability associated with the IDEF alert and it includes the CVE identifier.

The first components of Risk Manager that support CVE today are Web IDS and NSA.

B.4 How you can benefit from CVE

Consumers of security information

The CVE name will give you a standardized identifier for any given vulnerability or exposure. Knowing this identifier will allow you to quickly and accurately access information about the problem across multiple information sources that are CVE-compatible. For example, when a security tool's reports contain references to CVE names, you may then access fix information in a separate CVE-compatible database.

CVE may also benefit consumers of security information by facilitating better research on vulnerabilities.

Producers of security information

By mapping your own vulnerability related information to CVE, you can dramatically increase the value of your information to consumers by providing them with the ability to relate it to other CVE-compatible data sources. The mappings will also allow you to share and compare your information with other producers.

Appendix C. Special notices

This publication is intended to help IBM and Tivoli customers, Business Partners and Professionals to understand, plan and deploy the Tivoli SecureWay Risk Manager product and infrastructure. The information in this publication is not intended as the specification of any programming interfaces that are provided by Tivoli SecureWay Risk Manager. See the Publications section of the IBM Programming Announcement for Tivoli SecureWay Risk Manager for more information about what publications are considered to be product documentation.

References in this publication to IBM products, programs, or services do not imply that IBM intends to make these available in all countries in which IBM operates. Any reference to an IBM product, program, or service is not intended to state or imply that only IBM's product, program, or service may be used. Any functionally equivalent program that does not infringe any of IBM's intellectual property rights may be used instead of the IBM product, program, or service.

Information in this book was developed in conjunction with use of the equipment specified, and is limited in application to those specific hardware and software products and levels.

IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to the IBM Director of Licensing, IBM Corporation, North Castle Drive, Armonk, NY 10504-1785.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact IBM Corporation, Dept. 600A, Mail Drop 1329, Somers, NY 10589 USA.

Such information may be available, subject to appropriate terms and conditions including, in some cases, payment of a fee.

The information contained in this document has not been submitted to any formal IBM test and is distributed AS IS. The use of this information or the implementation of any of these techniques is a customer responsibility and depends on the customer's ability to evaluate and integrate them into the customer's operational environment. While each item may have been reviewed by IBM for accuracy in a specific situation, there is no guarantee

that the same or similar results will be obtained elsewhere. Customers attempting to adapt these techniques to their own environments do so at their own risk.

Any pointers in this publication to external Web sites are provided for convenience only and do not in any manner serve as an endorsement of these Web sites.

The following terms are trademarks of the International Business Machines Corporation in the United States and/or other countries:

IBM ®	AIX
Lotus	Netfinity
OS/390	RACF
Redbooks	Redbooks Logo 
RS/6000	SecureWay
SP	SP2
WebIDS	XT

The following terms are trademarks of other companies:

Tivoli, Manage. Anything. Anywhere.,The Power To Manage., Anything. Anywhere.,TME, NetView, Cross-Site, Tivoli Ready, Tivoli Certified, Planet Tivoli, and Tivoli Enterprise are trademarks or registered trademarks of Tivoli Systems Inc., an IBM company, in the United States, other countries, or both. In Denmark, Tivoli is a trademark licensed from Kjøbenhavns Sommer - Tivoli A/S.

Cisco SecureIDS and NetRanger are trademarks of Cisco Systems, Inc.

ISS RealSecure is a trademark of Internet Security Systems, Inc.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States and/or other countries.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States and/or other countries.

PC Direct is a trademark of Ziff Communications Company in the United States and/or other countries and is used by IBM Corporation under license.

UNIX is a registered trademark in the United States and other countries licensed exclusively through The Open Group.

Other company, product, and service names may be trademarks or service marks of others.

Appendix D. Related publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this redbook.

D.1 IBM Redbooks

For information on ordering these publications see “How to get IBM Redbooks” on page 215.

- *Understanding IBM SecureWay FirstSecure*, SG24-5498
- *Deploying a Public Key Infrastructure*, SG24-5512
- *LDAP Implementation Cookbook*, SG24-5110
- *Understanding LDAP*, SG24-4986
- *IP Network Design Guide*, SG24-2580
- *A Secure Way to Protect Your Network: IBM SecureWay*, SG24-5855
- *Tivoli SecureWay Policy Director, Centrally Managing e-business Security,* SG24-6008
- *Tivoli Framework 3.7 User’s Guide*, GC31-8433
- *Tivoli TEC 3.7 Adaptor’s Guide*, GC32-0668
- *Tivoli TEC 3.7 User’s Guide*, GC32-0667

D.2 IBM Redbooks collections

Redbooks are also available on the following CD-ROMs. Click the CD-ROMs button at ibm.com/redbooks for information about all the CD-ROMs offered, updates and formats.

CD-ROM Title	Collection Kit Number
IBM System/390 Redbooks Collection	SK2T-2177
IBM Networking Redbooks Collection	SK2T-6022
IBM Transaction Processing and Data Management Redbooks Collection	SK2T-8038
IBM Lotus Redbooks Collection	SK2T-8039
Tivoli Redbooks Collection	SK2T-8044
IBM AS/400 Redbooks Collection	SK2T-2849
IBM Netfinity Hardware and Software Redbooks Collection	SK2T-8046
IBM RS/6000 Redbooks Collection	SK2T-8043
IBM Application Development Redbooks Collection	SK2T-8037
IBM Enterprise Storage and Systems Management Solutions	SK3T-3694

D.3 Other resources

These publications are also relevant as further information sources:

- Christian Gigandet, *Integration of Host-based Intrusion Detection Systems into the Tivoli Enterprise Console*, RZ3253 (#93299) 06/26/2000, IBM Research, Zurich Research Laboratory, 8803 Rueschlikon, Switzerland
- *Risk Manager Installation and User's Guide, Version 1.0*
- G. Malkin, "Rip Version 2," *Request for Comments (Standards Track) 2453,* Internet Engineering Task Force, November 1998.
- J, Case, R. Mandy, D, Partain, and B. Stewart, "Introduction to Version 3 of teh Internet-Standard Network Management Framework," *Request for Comments (Informational) 2570*, Internet Engineering Task Force, April 1999.

D.4 Referenced Web sites

These Web sites are also relevant as further information sources:

- `http://www.tivoli.com`

  Tivoli's homepage provides you with the best entry point to all of Tivoli's Management software.

- `http://www.tivoli.com/products/index/secureway_risk_mgr/`

  This is the first stop for all Tivoli SecureWay Risk Manager related information. Here, you will find all sorts of product documentation and demos as well as product updates and fixes.

- `http://www.cve.mitre.org`

  This is the CVE Web site, which holds information about CVE itself, the CVE list, and a wealth of links to related information and activities.

- `http://csrc.nist.gov/icat/`
  `http://www.securityfocus.com`

  These are sites that were searchable with CVE names and provided useful information as well as direct links to other providers' records.

How to get IBM Redbooks

This section explains how both customers and IBM employees can find out about IBM Redbooks, redpieces, and CD-ROMs. A form for ordering books and CD-ROMs by fax or e-mail is also provided.

- **Redbooks Web Site** ibm.com/redbooks

  Search for, view, download, or order hardcopy/CD-ROM Redbooks from the Redbooks Web site. Also read redpieces and download additional materials (code samples or diskette/CD-ROM images) from this Redbooks site.

  Redpieces are Redbooks in progress; not all Redbooks become redpieces and sometimes just a few chapters will be published this way. The intent is to get the information out much quicker than the formal publishing process allows.

- **E-mail Orders**

  Send orders by e-mail including information from the IBM Redbooks fax order form to:

	e-mail address
In United States or Canada	pubscan@us.ibm.com
Outside North America	Contact information is in the “How to Order” section at this site: http://www.elink.ibmlink.ibm.com/pbl/pbl

- **Telephone Orders**

United States (toll free)	1-800-879-2755
Canada (toll free)	1-800-IBM-4YOU
Outside North America	Country coordinator phone number is in the “How to Order” section at this site: http://www.elink.ibmlink.ibm.com/pbl/pbl

- **Fax Orders**

United States (toll free)	1-800-445-9269
Canada	1-403-267-4455
Outside North America	Fax phone number is in the “How to Order” section at this site: http://www.elink.ibmlink.ibm.com/pbl/pbl

This information was current at the time of publication, but is continually subject to change. The latest information may be found at the Redbooks Web site.

IBM Intranet for Employees

IBM employees may register for information on workshops, residencies, and Redbooks by accessing the IBM Intranet Web site at http://w3.itso.ibm.com/ and clicking the ITSO Mailing List button. Look in the Materials repository for workshops, presentations, papers, and Web pages developed and written by the ITSO technical professionals; click the Additional Materials button. Employees may access MyNews at http://w3.ibm.com/ for redbook, residency, and workshop announcements.

IBM Redbooks fax order form

Please send me the following:

Title	Order Number	Quantity

First name	Last name	
Company		
Address		
City	Postal code	Country
Telephone number	Telefax number	VAT number

☐ Invoice to customer number _____

☐ Credit card number _____

Credit card expiration date	Card issued to	Signature

We accept American Express, Diners, Eurocard, Master Card, and Visa. Payment by credit card not available in all countries. Signature mandatory for credit card payment.

Glossary

APAR	Authorized Program Analysis Report
B2B	Business to Business
B2C	Business to Consumer
BAROC	Basic Recorder of Objects in C
BID	Numbering scheme for vulnerabilities used by BUGTRAQ.
buffer overflow	A special type of vulnerability in software programs that allow a hacker to gain unauthorized access to parts of a system's memory.
BUGTRAQ	Mailing list based service that updates subscribers about new vulnerabilities and security breaches.
CAT	Common Adapter Toolkit
CERT	CERT coordination center, reporting center for Internet security problems. CERT is a registered service mark of Carnegie Mellon University (not an acronym).
CVE	Common Vulnerabilities and Exposures
directive	Configuration or control statement in a NSA control file.
ERS	IBM Emergency Response Service
event adapter	A single module that transports events generated by supported IDS applications directly to the TEC server for processing.
FID	finding identifier
findings	Any information gathered by a scan.
IDEF	Intrusion Detection Exchange Format
IDWG	Intrusion Detection Working Group
libc	Name of the C compiler library package under Linux.
logfile adapter	Collect events out of the NT event log or the UNIX syslog at certain intervals and send them to the TEC server for processing.
man pages	Online help library under UNIX.
MITRE	The MITRE Corporation (www.mitre.org).
MSKB	Microsoft Knowledge Base
network scan	Activity to gain information about a network and the systems attached to it
NSA	Network Services Auditor
PERL	A scripting language that is popular in UNIX
PKCS	Public Key Cryptography Standards
PKI	Public Key Infrastructure

pre-adapter	Retrieves and formats event information from third party products like Cisco Secure IDS or ISS Realsecure and stores this information in either a syslog format or NT event log format
RedHat	A Linux distributor
SuSE	A Linux distributor
TEC	Tivoli Enterprise Console
TME	Tivoli Management Environment
vi	Text editor, widely used under UNIX
XF	X-Force Vulnerability database service offered by ISS

Index

F

G

H

I

J

L

N

O

P

R

S

T

U

V

W

IBM Redbooks review

Your feedback is valued by the Redbook authors. In particular we are interested in situations where a Redbook "made the difference" in a task or problem you encountered. Using one of the following methods, **please review the Redbook, addressing value, subject matter, structure, depth and quality as appropriate.**

- Use the online **Contact us** review redbook form found at **ibm.com**/redbooks
- Fax this form to: USA International Access Code + 1 914 432 8264
- Send your comments in an Internet note to redbook@us.ibm.com

Document Number **Redbook Title**	SG24-6021-00 Tivoli SecureWay Risk Manager Correlating Enterprise Risk Management
Review	
What other subjects would you like to see IBM Redbooks address?	
Please rate your overall satisfaction:	O Very Good O Good O Average O Poor
Please identify yourself as belonging to one of the following groups:	O Customer O Business Partner O Solution Developer O IBM, Lotus or Tivoli Employee O None of the above
Your email address: The data you provide here may be used to provide you with information from IBM or our business partners about our products, services or activities.	O Please do not use the information collected here for future marketing or promotional contacts or other communications beyond the scope of this transaction.
Questions about IBM's privacy policy?	The following link explains how we protect your personal information. **ibm.com**/privacy/yourprivacy/

(0.2”spine)
0.17”<->0.473”
90<->249 pages

Tivoli SecureWay Risk Manager

Correlating Enterprise Risk Management

Comprehensive coverage of the correlation engine methodology

Integrating new applications into the Risk Manager framework

Complete introduction to the Network Services Auditor

Tivoli SecureWay Risk Manager is an integrated e-business security management solution based on the Tivoli Framework. It enables customers to centrally manage attacks, threats, and vulnerabilities by correlating security alerts from diverse security point product deployments. By correlating these security alerts across intrusion detectors and Web servers, Risk Manager enables administrators to eliminate clutter such as false-positives, centrally correlate distributed attacks, identify real security threats with a high degree of integrity and confidence, and use adaptive event response using the Tivoli Enterprise Console.

This redbook explains the architecture and correlation engine of the Tivoli SecureWay Risk Manager. It describes the details of the event class structure and helps you integrate new application sensors. The Network Services Auditor (NSA) extends the scope of Risk Manager by providing a vulnerability scanner for network resources. It will be technically described in this redbook for the first time.

This book is a valuable resource for security administrators and architects who wish to understand and implement a centralized risk correlation mechanism.

INTERNATIONAL TECHNICAL SUPPORT ORGANIZATION

BUILDING TECHNICAL INFORMATION BASED ON PRACTICAL EXPERIENCE

IBM Redbooks are developed by the IBM International Technical Support Organization. Experts from IBM, Customers and Partners from around the world create timely technical information based on realistic scenarios. Specific recommendations are provided to help you implement IT solutions more effectively in your environment.

For more information:
ibm.com/redbooks

SG24-6021-00 ISBN 0738418064