

Meeting the Challenge of Log Management for Unix and Linux Systems

Written by
Randy Franklin Smith CEO, Monterey Technology Group, Inc.
Publisher of UltimateWindowsSecurity.com

© 2010 Quest Software, Inc.

ALL RIGHTS RESERVED.

This document contains proprietary information protected by copyright. No part of this document may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying and recording for any purpose without the written permission of Quest Software, Inc. ("Quest").

The information in this document is provided in connection with Quest products. No license, express or implied, by estoppel or otherwise, to any intellectual property right is granted by this document or in connection with the sale of Quest products. EXCEPT AS SET FORTH IN QUEST'S TERMS AND CONDITIONS AS SPECIFIED IN THE LICENSE AGREEMENT FOR THIS PRODUCT, QUEST ASSUMES NO LIABILITY WHATSOEVER AND DISCLAIMS ANY EXPRESS, IMPLIED OR STATUTORY WARRANTY RELATING TO ITS PRODUCTS INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT. IN NO EVENT SHALL QUEST BE LIABLE FOR ANY DIRECT, INDIRECT, CONSEQUENTIAL, PUNITIVE, SPECIAL OR INCIDENTAL DAMAGES (INCLUDING, WITHOUT LIMITATION, DAMAGES FOR LOSS OF PROFITS, BUSINESS INTERRUPTION OR LOSS OF INFORMATION) ARISING OUT OF THE USE OR INABILITY TO USE THIS DOCUMENT, EVEN IF QUEST HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. Quest makes no representations or warranties with respect to the accuracy or completeness of the contents of this document and reserves the right to make changes to specifications and product descriptions at any time without notice. Quest does not make any commitment to update the information contained in this document.

If you have any questions regarding your potential use of this material, contact:

Quest Software World Headquarters

LEGAL Dept

5 Polaris Way

Aliso Viejo, CA 92656

www.quest.com

E-mail: **legal@quest.com**

Refer to our Web site for regional and international office information.

Trademarks

Quest, Quest Software, the Quest Software logo, AccessManager, ActiveRoles, Aelita, Akonix, AppAssure, Benchmark Factory, Big Brother, BridgeAccess, BridgeAutoEscalate, BridgeSearch, BridgeTrak, BusinessInsight, ChangeAuditor, ChangeManager, Defender, DeployDirector, Desktop Authority, DirectoryAnalyzer, DirectoryTroubleshooter, DS Analyzer, DS Expert, Foglight, GPOAdmin, Help Desk Authority, Imceda, IntelliProfile, InTrust, Invirtus, iToken, IWatch, JClass, Jint, JProbe, LeccoTech, LiteSpeed, LiveReorg, LogAdmin, MessageStats, Monosphere, MultSess, NBSpool, NetBase, NetControl, Npulse, NetPro, PassGo, PerformaSure, Point,Click,Done!, PowerGUI, Quest Central, Quest vToolkit, Quest vWorkSpace, ReportAdmin, RestoreAdmin, ScriptLogic, Security Lifecycle Map, SelfServiceAdmin, SharePlex, Sitraka, SmartAlarm, Spotlight, SQL Navigator, SQL Watch, SQLab, Stat, StealthCollect, Storage Horizon, Tag and Follow, Toad, T.O.A.D., Toad World, vAutomator, vControl, vConverter, vFoglight, vOptimizer, vRanger, Vintela, Virtual DBA, VizionCore, Vizioncore vAutomation Suite, Vizioncore vBackup, Vizioncore vEssentials, Vizioncore vMigrator, Vizioncore vReplicator, WebDefender, Webthority, Xaffire, and XRT are trademarks and registered trademarks of Quest Software, Inc in the United States of America and other countries. Other trademarks and registered trademarks used in this guide are property of their respective owners.

February 2010

Contents

- Executive Summary3
- Native Auditing in Unix and Linux4
 - Capabilities4
 - Syslog.....4
 - Unix Auditing5
 - Linux Auditing.....6
 - Other Logs.....7
- Functionality Gaps Create Security and Compliance Problems.....7
 - Aggregation7
 - Archival.....8
 - Integrity.....8
 - Log File Modification* 8
 - Bypassed Audit Trails*..... 9
 - Alerting and Reporting9
- Meeting the Challenge with Quest InTrust.....10
 - InTrust Architecture.....10
 - Support for Unix and Linux10
 - Syslog Messages11
 - Audit Logs11
 - Text Logs.....12
 - Configuration Files12
 - Pre-Built and Custom Reports12
- Conclusion.....14
- About the Author15

Executive Summary

Unix and Linux generate a wide array of audit logs. Modern versions of Linux and Unix provide a formal audit system that creates a detailed audit trail of security activity across all of the operating system's components. When combined with legacy text-based and syslog-based audit trails, each Linux and Unix system can provide a wealth of audit data.

However, Unix and Linux audit logs vary greatly in terms of format, content and reliability, even within one flavor of Unix or distribution of Linux. Also, Unix and Linux auditing provides only some rudimentary log rotation and aggregation capabilities. To meet today's security and compliance requirements, organizations need a complete, secure enterprise log management and monitoring solution that includes data collection, real-time alerting, reporting, and secure archival.

Quest InTrust is an enterprise log management solution that enables organizations to comply with log management requirements in a heterogeneous environment. InTrust securely collects, stores, reports on, and alerts on event data from a wide variety of systems, including the most common flavors of Unix and Linux. Quest InTrust also provides operating system-specific agents for Solaris, HP-UX, AIX, Red Hat and SuSe that take into account the subtle differences and special capabilities unique to each flavor of Unix and Linux. Quest InTrust efficiently collects their voluminous raw audit trail data and refines it into easy-to-understand, actionable information. This white paper explains how Quest InTrust enables enterprises confronted with a variety of Unix, Linux, network devices and Windows systems—each with its own audit trail process and format—to securely monitor their systems and comply with log management and monitoring requirements.

Native Auditing in Unix and Linux

Capabilities

Unix audit capabilities have evolved over time, beginning from when security was no more than an afterthought. As intruders gained sophistication, administrators began relying on logging facilities that were initially conceived for occasional troubleshooting and analysis. In response to audit requirements largely mandated by the defense sector, Unix added a formal audit system that varies with each vendor's implementation.

Factors that contribute to the widely varying sources and formats of audit data include:

- The Unix's long evolution has yielded a succession of logs and audit trails as the operating system as has grown and security requirements have increased
- Unix/Linux's modular architecture means that each daemon and component generates its own audit data
- Unix's many flavors and Linux's open source freedom allow each vendor to implement auditing differently

Syslog

The syslog is a logging facility, network protocol, and source of audit data, though the latter is an unintended byproduct. Syslog in Unix and Linux is similar to the Event Logging Service in Windows; the syslog daemon (syslogd) allows any process on the system to send it log messages.¹

A key point to understand about syslog is that it is essentially a generic logging service—each program's developer gets to decide what events generate a log entry. Developers must classify messages according to several pre-defined facilities (listed in Table 1) to identify the source of the message, and seven levels to define the severity of the message. This means the developer determines the information that is reported in the log entry, the information's format, and if the user has any control over how many log messages are generated. This means syslog is not a comprehensive audit facility because coverage of security relevant events is inconsistent from one program to another.

Through its configuration file (/etc/syslog.conf), administrators control what syslogd does with the messages it receives. For logging purposes, the administrator can send messages to a local log file (usually under /var/log) or over the network to syslogd on another system. The criteria used to control what syslog does with messages are defined by the facility and level.

Unfortunately, the pre-defined facilities are not granular enough to properly differentiate between multiple sources of log messages. They are unable to make good decisions about whether the messages should be discarded or logged, and where they should be logged. Moreover, there are no clearly defined standards for syslog message levels. This means that no assumptions can be made about the message level as reported by different programs for different events. The only choice is to log everything, and then make filtering decisions after analyzing the messages generated by a given program.

By default, most Unix and Linux systems route syslog messages to a number of files in the /var/log subdirectory. The files differ depending on the operating system, flavor and version, but commonly include those in Table 2.

Instead of using local log files, administrators can configure syslogd to send messages with specified facilities and

Table 1: Syslog Facilities

- **kern** – Kernel
- **user** – Application or user processes
- **mail/news/UUCP/cron** –Subsystems
- **daemon** – System daemons
- **auth** – Authentication and authorization related commands
- **lpr** – Line printer spooling subsystem
- **mark** – Inserts timestamp into log data at regular intervals
- **local0-local7** – Eight facilities for customized auditing
- **syslog** – Internal messages generated by syslog itself
- **authpriv** – Non-system authorization messages

¹ For more information about syslog, see <http://www.precision-guesswork.com/sage-guide/syslog-overview.html>.

levels over the network to another computer's syslog daemon using the syslog protocol,² usually on UDP port 514. At the destination syslogd, any messages received from remote systems are processed according to the local syslog.conf file's rules. This provides the administrator with some rudimentary log aggregation capabilities.

At some point, logs need to be rotated and archived. The logrotate³ command in Unix and Linux provides a way to rotate log files, move them to an archive directory, and optionally compress and or mail the logs.

```
Dec 6 14:57:18 localhost sshd[2352]: Server listening on :: port 22.
Dec 6 14:57:18 localhost sshd[2352]: error: Bind to port 22 on 0.0.0.0 failed:
Address already in use.
Dec 6 14:59:22 localhost login: pam_unix(login:session): session opened for use
r root by (uid=0)
Dec 6 14:59:22 localhost login: ROOT LOGIN ON tty1
Dec 6 15:27:33 localhost sshd[2352]: Received signal 15: terminating.
Dec 6 15:27:36 localhost sshd[3902]: Server listening on :: port 22.
Dec 6 15:27:36 localhost sshd[3902]: error: Bind to port 22 on 0.0.0.0 failed:
Address already in use.
Dec 6 15:29:12 localhost login: pam_unix(login:session): session closed for use
r root
Dec 6 15:29:15 localhost sshd[3902]: Received signal 15: terminating.
Dec 6 15:30:41 localhost sshd[2341]: Server listening on :: port 22.
Dec 6 15:30:41 localhost sshd[2341]: error: Bind to port 22 on 0.0.0.0 failed:
Address already in use.
Dec 6 15:32:42 localhost login: pam_unix(login:session): session opened for use
r root by (uid=0)
Dec 6 15:32:42 localhost login: ROOT LOGIN ON tty1
Dec 6 15:35:44 localhost login: pam_unix(login:session): session closed for use
r root
Dec 6 15:35:46 localhost sshd[2341]: Received signal 15: terminating.
Dec 6 16:02:28 localhost sshd[2345]: Server listening on :: port 22.
Dec 6 16:02:28 localhost sshd[2345]: error: Bind to port 22 on 0.0.0.0 failed:
```

Example /var/log/secure log from CentOS

Unix Auditing

In this paper, Solaris Basic Security Module (BSM) serves as the basis for discussing Unix audit logs, but other flavors of Unix provide similar capabilities, such as the HP-UX Auditing System and AIX's audit subsystem.

As explained earlier, syslog is a logging facility with no standards for logging activities or how messages are formatted. BSM, on the other hand, is a formal audit facility with defined standards regarding what is logged and the format of the log entries. The kernel traps most auditable events and logs the messages to buffers where they can be retrieved by the auditd daemon for output to the current binary audit log. User processes can also report security events through the audit system.

Every event type has a unique event ID number, with 0-2047 reserved for kernel messages, 2048-

Table 2: Common syslog Log Files in /var/log

- **auth.log** – Authentication information
- **boot.log** – Boot information
- **crond** – Scheduled cron job events
- **daemon.log** – Daemon messages
- **dmesg** – Kernel messages
- **errors.log** – Error messages
- **everything.log** – Catch-all log
- **httpd** – Apache messages
- **mail.log** – Mail server logs
- **messages.log** – General messages
- **mysqld.log** – MySQL database log
- **secure** – Security log (but far from comprehensive)
- **syslog.log** – A log for the log system
- **vsftpd.log** – FTP server, vsftpd

² See IETF RFC5424 - The Syslog Protocol at <http://tools.ietf.org/html/rfc5424#section-5>.

³ See http://linuxcommand.org/man_pages/logrotate8.html for a good explanation of logrotate.

32767 reserved for messages from Sun OS user-level programs, and 32768-65536 reserved for third-party programs. Events are categorized according to *audit class*: an administrator can control which security events are audited by assigning different events to different audit classes and then enabling each class for successful and/or failed events, or for no auditing at all. Administrators configure these rules in the `/etc/security/audit_control` file, which define machine-wide audit defaults. Administrators can refine those defaults on a user-by-user basis by defining audit flags for specific users in the `/etc/security/audit_user` file.

Auditd writes audited security events to audit log files in the directories specified in the `audit_control` file, and rotates to new files and switches directories based upon other `audit_control` parameters. The audit log file naming convention ensures the file's name includes the time period it covers, the host name of the system, and a number that signifies the file's order in the rotation.

Because of the number of generated files, the huge amount of data in each one, and their binary format, audit log files are not intended to be reviewed by administrators. To create usable information, BSM provides the `auditreduce` and `praudit` commands. With `auditreduce`, administrators can merge multiple audit logs while simultaneously selecting a subset of audit records based on time, date, users, and/or audit class. Then, administrators use the `praudit` command to translate the output from `auditreduce` into a format that can be easily read. You can see in the example below, event `praudit` massaged audit data is difficult to read and cryptic.

```
header,144,2,fcntl(2),,Thu Aug 10 22:01:22 2000, + 60005500 msec
argument,2,0x1,cmd
path,/devices/pseudo/cn@0:console
attribute,20620,root,TTY,8388632,154849,0
subject,root,root,other,root,other,194,191,0 0 log1
return,success,0
header,122,2,fcntl(2),,Thu Aug 10 22:01:22 2000, + 70007000 msec
argument,2,0x1,cmd
argument,1,0x1,no path: fd
attribute,10000,root,other,48234496,46,0
subject,root,root,other,root,other,196,191,0 0 log1
return,success,0
header,139,2,execve(2),,Thu Aug 10 22:01:22 2000, + 80008500 msec
path,/usr/bin/grep
attribute,100555,root,bin,8388632,15243,0
exec_args,2,
/usr/bin/grep,DEF_UMASK=
subject,root,root,other,root,other,196,191,0 0 log1
return,success,0
header,122,2,fcntl(2),,Thu Aug 10 22:01:22 2000, + 130007000 msec
argument,2,0x1,cmd
argument,1,0x0,no path: fd
attribute,10000,root,other,48234496,47,0
subject,root,root,other,root,other,194,191,0 0 log1
return,success,0
header,164,2,execve(2),,Thu Aug 10 22:01:22 2000, + 180001000 msec
path,/usr/bin/sed
attribute,100555,root,bin,8388632,15323,0
exec_args,2,
/usr/bin/sed,s/^.*DEF_UMASK=\([0-9]\{1,\}\)\.*/\1/
subject,root,root,other,root,other,194,191,0 0 log1
return,success,0
header,144,2,fcntl(2),,Thu Aug 10 22:01:22 2000, + 250003500 msec
argument,2,0x1,cmd
path,/devices/pseudo/cn@0:console
attribute,20620,root,TTY,8388632,154849,0
subject,root,root,other,root,other,197,191,0 0 log1
return,success,0
```

Solaris BSM Audit Log Example

Most Unix implementations of binary audit logging do not support streaming of events to other programs for real-time analysis and processing. AIX's auditing subsystem is a notable exception; its stream mode allows log management and intrusion detection systems to consume audit events as they occur.

Linux Auditing

All distributions of Linux that are based on the 2.6 kernel provide a consistent and powerful audit system with some similarities to Unix binary audit logging. Like Unix, the Linux kernel traps auditable events and writes them to buffers where they are processed by the `auditd` daemon. Without any special configuration,

some kernel components and modules automatically report security activity to the audit log. For instance, the pluggable authentication module (PAM) system of Linux reports login-related events to the audit log.

Beyond this basic logging, administrators control what events and activity are audited via the `/etc/audit/audit.rules` file and the `auditctl` command. Unlike Unix, Linux does not use the concept of audit classes, but administrators can enable or disable auditing for each system call and can define whether, or how, system calls are audited by specifying criteria pertaining to entry parameters and exit result codes. By defining system call audit entries for file operations such as open, read or write, administrators can audit system-wide file system activity, but that may create massive amounts of audit log data in a short amount of time. Administrators can define more granular file auditing through watch rules, which limit auditing to a specified file or directory. Watch rules can be further refined to audit certain types of access.

Administrators control the location of the audit log file and define log rotation behavior in `/etc/audit/auditd.conf`. The `aureport` command provides summary statistics of the audit log, and `ausearch` allows administrators to query one or more audit logs for a multitude of selection criteria.

In addition to sending audit events to a traditional log file, Linux also provides an extensible audit dispatcher (`audispd`) that allows programs to receive and process audit events as they occur. Standard `audispd` plug-ins include the ability to send audit events to `syslog` or over the network to the audit daemon on another system.

Other Logs

There are programs that report security-related events to a log file other than `syslog` or native audit logs. Specific examples of “other” log files are beyond the scope of this paper, but it’s important to note that such text based log files exist and may need to be managed.

Functionality Gaps Create Security and Compliance Problems

Log generation, however, is just the beginning of the log management process required for security and compliance. Although Unix and Linux are capable of generating large amounts of detailed audit information, they fall short in other log management requirements. For instance, NIST Special Publication 800-92, “Guide to Computer Security Log Management,” which provides guidance for FISMA compliance, mandates that organizations “create and maintain a secure log management infrastructure.”

Comprehensive log management means logs must be collected to a central aggregation point for secure archiving, monitoring, and reporting. Without secure log management, organizations cannot follow proven security best practices or comply with regulatory and industry requirements. These requirements are common to all compliance regulations, but in this paper we will cite examples from the Payment Card Industry (PCI) Data Security Standard and the NIST Special Publication 800-92 because of their clear and straightforward language.

Aggregation

“Promptly back-up audit trail files to a centralized log server.” – PCI DSS 10.5.3

Logs “should be ... transmitted to the log management infrastructure.” – NIST SP 800-92

The first step in comprehensive log management is aggregating logs to a secure, central location. Unix and Linux provide some log aggregation functionality, but it has some important gaps that lead to security and compliance problems.

Table 3: Log Management Requirements in Common Compliance Regulations

HIPAA – See “[HIPAA Security Compliance Project – Identification of Logging and Auditing Requirements](#)” from SANS

SOX – See “DS5.5 Security Testing, Surveillance and Monitoring” in [COBIT 4.1](#).

FISMA – See [NIST Special Publication 800-92: “Guide to Computer Security Log Management”](#)

PCI – See “Requirement 10: Track and monitor all access to network resources and cardholder data” in [PCI Data Security Standard](#)

At first blush, syslog would seem to provide out-of-the-box aggregation because of its ability to direct syslog messages over the network to a central syslog server. However, syslog is widely recognized to be unreliable and insecure. Syslog provides no protection against:

- Lost messages
- Modification or deletion of messages while in transit over the network
- Spoofed messages

A number of replacements for syslog, such as syslog-ng and rsyslog, have had some success, but none has become the de facto standard. Moreover, the syslog data is inadequate as an audit trail even before it traverses the network, since any process can report any type of message to the syslog daemon. Furthermore, while syslog does provide some valuable coverage of security activity, it is by no means comprehensive.

Aggregation for Unix and Linux audit systems is a mixed bag. The Linux audit system provides audisp-remote, which supports secure transmission of audit messages from many Linux systems to the audit daemon on a central server. At this time, however, little post-aggregation log management is provided beyond the standard log rotation features of Linux auditing and the aureport and ausearch commands. Unix audit systems do not provide any log aggregation on the level of Linux's audisp-remote.

Many Unix and Linux administrators are skilled scripters and may be tempted to take a "roll your own" approach to log aggregation through a collection of cron jobs and scripts that move log files to a central repository. But separation of duty and log integrity requirements require a degree of health monitoring and documentation; it is impractical for individual system administrators to cooperatively fulfill these requirements through scripts and scheduled tasks.

Archival

"Retain audit trail history for at least one year." – PCI DSS

"Logs are retained for the required period of time." – NIST 800-92

Security investigations and compliance requirements demand that organizations be able to produce and analyze logs from months or even years ago. Log rotation features of syslog and Unix and Linux audit systems are appropriate for single-system, low-security settings. Perhaps when combined with the basic aggregation features of syslog, Linux's audisp-remote, and some auditreduce scripts for Unix systems, a workable archival solution could be cobbled together for a handful of systems.

However, for enterprises with hundreds or thousands of systems, the daily burden of thousands of logs and gigabytes of data quickly overwhelms such a scripted approach. Instead, these organizations need efficient compression and centralized, automated handling of log files to prevent accidental deletion and ensure data integrity.

Integrity

"Secure audit trails so they cannot be altered." – PCI DSS

Logs "need to be protected from breaches of their confidentiality and integrity." – NIST 800-92

Data integrity is crucial to log management. Without it, the cause of and extent of intrusions cannot be determined, evidence is thrown out of court and prosecutions are lost. Organizations may incur penalties from regulatory bodies, including possible criminal liability for managers and corporate directors. Data integrity affects log management in two ways:

- Log file modification
- Bypassed audit trails

Log File Modification

To ensure integrity, logs must be moved from monitored systems to a separate archive isolated from the systems **and** the administrators being audited as frequently as possible. This not only helps to protect logs from outside intruders, but acts as a deterrent for administrators with unlimited authority.

After being aggregated to a central archive point, the logs still need to be protected. The archive system must have different administrators than the systems that generated the logs. This requirement argues against using in-house scripts to aggregate and archive logs.

Finally, the archive system itself must be secured against intrusion. Digital signatures are widely used to provide final assurance that log data has not been tampered with.

Bypassed Audit Trails

Linux and Unix store security critical data and configuration settings (such as user accounts) in text files, which can be directly accessed and modified, thus bypassing the commands that would normally generate audit trail data or syslog messages. Therefore, audit system watches must be configured for important security files to trap modifications at the file system. Unfortunately, file system auditing in Unix and Linux suffers from the same problem as Windows: the audit data generated is very low level and voluminous, reflecting the interaction between the program and the file system.

Fulfilling these data integrity requirements is beyond the scope of Unix and Linux native capabilities.

Alerting and Reporting

“Review logs for all system components at least daily ... alerting tools may be used to achieve compliance.” – PCI DSS

“Traditionally, most logs have not been analyzed in a real-time or near-real-time manner. Without sound processes for analyzing logs, the value of the logs is significantly reduced.”

– NIST SP 800-92

High-priority security events, such as suspicious configuration changes or audit policy modifications, require the security staff to be alerted as close to real-time as possible. Organizations with a handful of Unix or Linux systems and a talented scripter on staff might be able to accomplish limited alerting with scripts based on the tail command. But for organizations with many systems to manage, this scripting approach runs into the same management issues as log aggregation and archival.

Other events, such as new account creation and group membership changes, require the review of daily reports. Other security activity needs to be documented in reports that are kept on hand for audit purposes. Different reports need to go to different people, and those that require timely review for compliance purposes need basic workflow tracking to identify when a report was reviewed and by whom.

The automatic production and mailing of a few basic text reports is well within the capabilities of Unix and Linux through cron jobs, the environment's respective audit reporting command (e.g., ausearch, or auditreduce and praudit), and a parsing tool for syslog-generated logs. But more sophisticated reports require a relational database or formatting as html or spreadsheets. This is beyond the native capabilities of these utilities, as is report archival and review tracking.

Meeting the Challenge with Quest InTrust

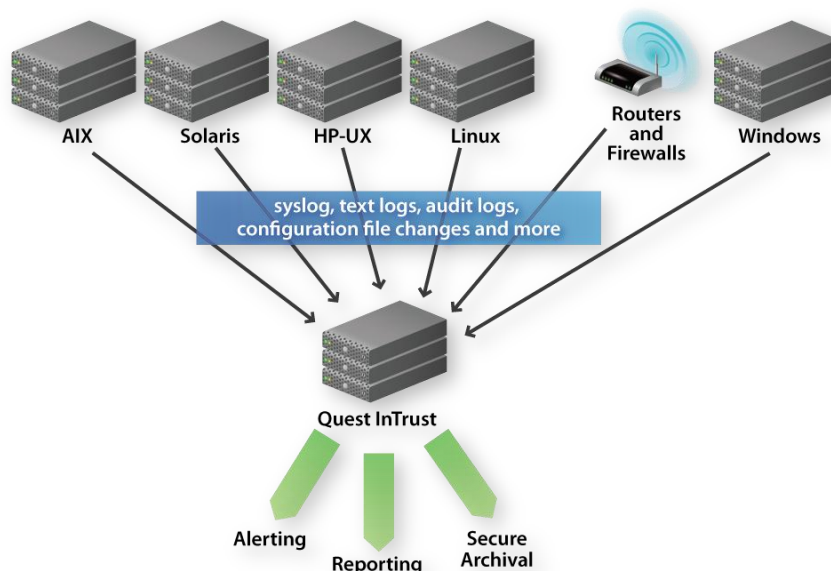
Quest InTrust is an enterprise log management solution that enables organizations to comply with log management requirements in a heterogeneous environment. InTrust securely collects, stores, reports on, and alerts on event data from a wide variety of systems, including the most common flavors of Unix and Linux.

Quest InTrust also provides operating system-specific agents for Solaris, HP-UX, AIX, Red Hat and SuSe that take into account the subtle differences and special capabilities unique to each flavor of Unix and Linux. For instance, the InTrust agent for AIX uses the unique stream mode of the AIX audit subsystem to efficiently process audit log events as they occur instead of in batches as binary log files are rotated.

Let's look at a brief overview of InTrust's architecture followed by a detailed description of InTrust's specialized support for Linux and Unix.

InTrust Architecture

With its SecureCollect technology, InTrust automates the secure collection of event logs. Automated collection schedules reduce IT workload and ensures logs are collected to a secure archive. InTrust agents improve the efficiency and security of log collection by filtering noise events on the local system and sending log data to the central archive via an authenticated, encrypted channel.



InTrust utilizes a unique, two-tiered storage architecture known as StoreMore. The first tier of StoreMore is the repository, which offers unparalleled long-term

compression. The second tier of StoreMore is the database, which enables advanced analysis and reporting. Most installations initially collect data to the repository and then move subsets to the database when needed. This unique storage architecture gives you better historical insight into your network than was previously possible.

InTrust's FlexReport technology allows organizations to create and distribute reports automatically. Staff can read reports online, create .PDFs for e-mailing, or post reports to an intranet or to SharePoint portal server. This enables organizations to track and document that reports were reviewed and acted upon.

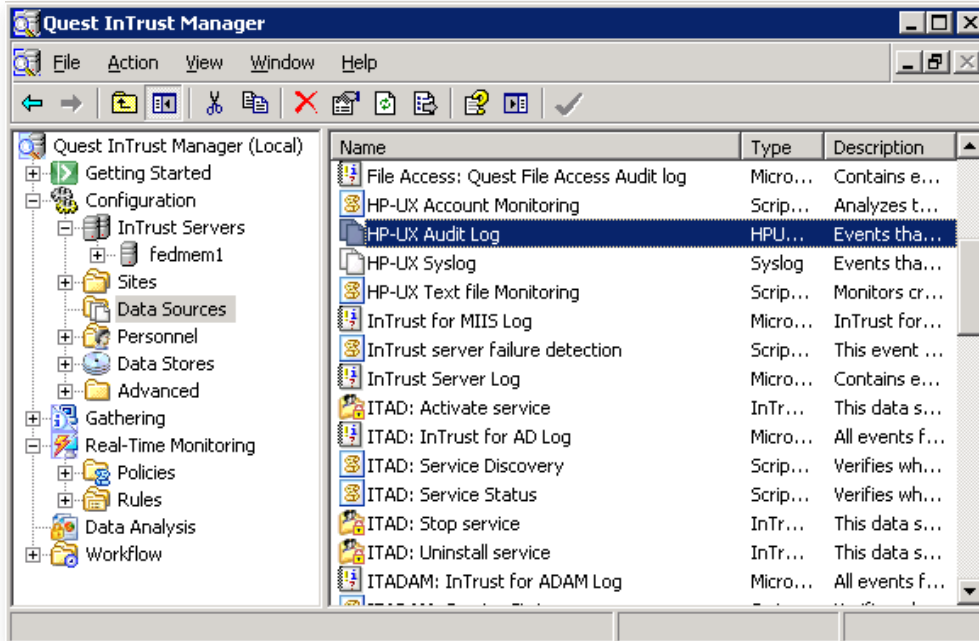
InTrust improves security through real-time alerts of unusual user and administrator activity, such as attempts to access files during off hours, multiple failed logon attempts followed by a successful log on, and other business-critical security events.

Support for Unix and Linux

To fulfill the security and compliance requirements of log management within each flavor of Unix and Linux, InTrust provides operating system-specific agents that support collection of syslogs, audit logs, and other text logs, as well as monitors configuration files that are independent of native commands or daemons.

Syslog Messages

As we saw earlier, syslog is insecure, but it remains a critical component for auditing and monitoring the Unix and Linux environment. Quest InTrust solves the syslog's reliability and security problems without the installation, configuration and management burden of many syslog replacements.



Quest InTrust Manager

in theory, InTrust provides a syslog listener service that could be used to aggregate Unix and Linux syslog messages; however, this listener is better suited to aggregating logs from network devices, such as routers, that cannot run third-party agents. The local Unix or Linux InTrust agent collects syslog messages and forwards them to the InTrust server via a secure channel. This protects log data from spoofing or tampering. The local agent also filters unwanted messages before forwarding data to the InTrust server, which reduces network load and improves scalability.

If network problems temporarily prevent delivery of syslog messages to the InTrust server, the local agent stores the messages in a secure cache until connectivity is restored. Thus audit trail continuity is protected from connectivity issues, as well as any erasure of the normal system logs by intruders.

The InTrust agent also provides real-time monitoring of syslog messages based on alert criteria pushed down from the central InTrust server. This allows organizations to respond quickly to suspicious security activity.

Audit Logs

Each flavor of Unix and Linux implements auditing differently. Therefore, each InTrust agent supports the audit system specific to its environment using the correct commands and syntax. The central InTrust server supports the file formats of Unix binary audit logs and provides secure, compressed archival as well as pre-built reports for each flavor of Unix and Linux. Each InTrust agent adheres to the centrally-defined collection policy, ensuring that audit logs are securely collected to the central archive consistently across all Unix and Linux computers in the organization. This relieves administrators from:

- Learning much of the intricacies of each system's audit system
- Ensuring audit policy is configured consistently across multiple systems and versions of Unix
- Writing different scripts to rotate and aggregate logs for each version of Unix and Linux
- Scheduling and monitoring the execution of log rotation and aggregation scripts across many systems

InTrust's support for Unix auditing offers the following benefits:

- Administrators can focus on managing systems instead of rotating logs.
- Security staff can respond to security incidents instead of collecting and archiving log files.
- The organization can meet compliance and security requirements for log management without wasting resources.

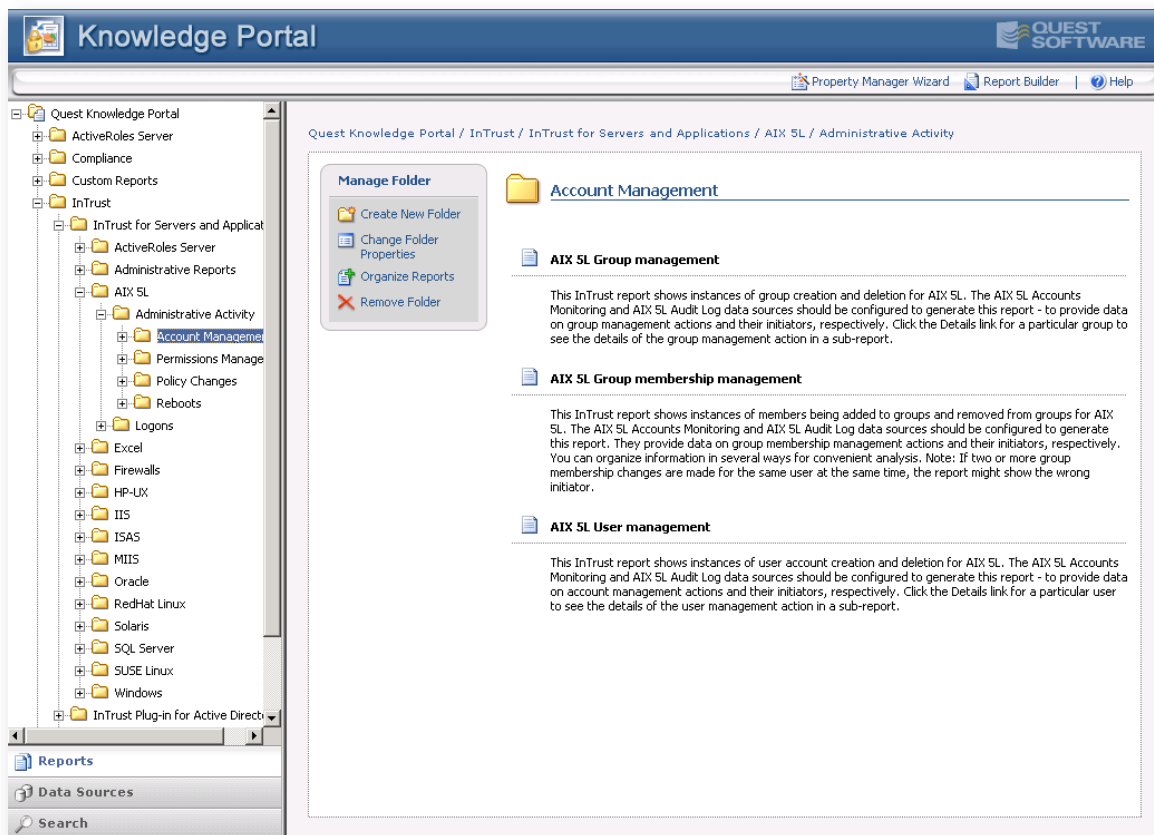
Text Logs

Some applications record important security events in their own text logs. This can result in crucial audit trails being outside the operating system's native syslog and audit logs.

InTrust allows administrators to define centralized policies that direct local InTrust agents to collect and securely transmit any text log file to the InTrust server for archiving, as well as for reporting using InTrust's custom report builder. This ensures that important application-generated audit trails are not missed and benefit from the organization's comprehensive log management infrastructure.

Configuration Files

Configuration files are the heart of Unix and Linux security, but they are vulnerable to direct modification that circumvents configuration change messages sent by commands and daemons to syslog or other logs.




Quest Knowledge Portal

InTrust allows organizations to centrally define monitoring policies that are implemented by local InTrust agents to detect changes in important configuration files, regardless of the method used to modify the file. Logs of these changes are aggregated to the InTrust server, and agents can generate real-time alerts based on centrally-defined monitoring policies. This ensures that organizations have a high-integrity audit trail of configuration files and the security staff receives immediate alerts when changes are made to the system.

Pre-Built and Custom Reports

InTrust provides a wealth of pre-built reports for each flavor of Unix and Linux; security staff can create new or customized reports to create reports specific to their environment. Compare the example InTrust Unix and Linux reports on this page to the raw logs samples included above. InTrust can deliver reports in a variety of formats including PDF and Excel.



AIX 5L Multiple Logon Failures

Filter for	Operation	Values
Number of Attempts:	Parameter expression	10
Time Interval:	Parameter expression	1
Measurement Units:	Parameter expression	hour

Host spb9487

User db2inst

From	To	Count
9/11/2007 2:58:58 AM	9/11/2007 2:59:20 AM	20

User UNKNOWN_USER

From	To	Count
7 3:00:39 AM		40
7 2:59:24 AM		40

chown command usage

This InTrust report shows usage of the chown command, which changes the owner of a file. Tracking this activity can

Filter for	Operation	Values						
Data Regions within table/matrix cells are ignored.								
User	UID	File Name	Old UID	New UID	Old GID	New GID	Result	Time
brian	999	///var/adm/sulog	0	0	0	0	Success	7/11/2007 2:07:59 PM
brian	999	/usr/lib/brian/rocket_codes	999	0	999	999	Failure	7/11/2007 11:50:29 AM
brian	999	/usr/lib/brian/rocket_codes	999	0	999	4294967295	Failure	7/11/2007 11:51:20 AM
root	0	/usr/lib/brian/rocket_codes	999	0	999	4294967295	Success	7/11/2007 11:51:47 AM
root	0	/usr/lib/brian/rocket_codes	0	999	999	4294967295	Success	7/11/2007 11:52:06 AM
brian	999	/usr/lib/brian/rocket_codes	999	65534	999	4294967295	Failure	7/11/2007 11:52:49 AM
brian	999	/usr/lib/brian/rocket_codes	999	0	999	4294967295	Failure	7/11/2007 12:06:06 PM
brian	999	/usr/lib/brian/rocket_codes/secure_1.txt	999	999	999	4294967295	Success	7/11/2007 12:06:43 PM
brian	999	/usr/lib/brian/rocket_codes/secure_1.txt	999	0	999	0	Failure	7/11/2007 12:07:09 PM
brian	999	/usr/lib/brian/rocket_codes/secure_1.txt	999	65534	999	4294967295	Failure	7/11/2007 12:07:22 PM
root	0	/usr/lib/brian/rocket_codes/secure_1.txt	999	65534	999	4294967295	Success	7/11/2007 12:08:30 PM
brian	999	/usr/lib/brian/rocket_codes/secure_1.txt	65534	999	999	4294967295	Failure	7/11/2007 12:09:04 PM
root	0	/usr/lib/brian/rocket_codes/secure_1.txt	65534	999	999	4294967295	Success	7/11/2007 12:09:35 PM

Conclusion

Unix and Linux generate a wealth of audit trail data, but its value is limited because of its quantity, the wide variety of sources and formats of log data and the raw, unrefined level of content. Moreover, there is no way to efficiently and securely collect, alert on, report on, and archive Unix and Linux audit data.

Quest InTrust solves these problems for the heterogeneous enterprise network, as well as specifically for Unix and Linux. To help manage the network, Quest InTrust uses agent-optional technology to consolidate log data from a wide variety of systems to a central point where the aggregated log data can be analyzed and securely stored for long-term archiving. To address the specific challenges presented by multiple flavors of Unix and Linux, Quest InTrust provides operating system-specific agents for Solaris, HP-UX, AIX, Red Hat and SuSe that take into account their subtle differences and special capabilities. Moreover, InTrust provides a comprehensive set of pre-built reports that intelligently refine raw log data into actionable information.

Quest InTrust enables enterprises confronted with a variety of Unix, Linux, network devices and Windows systems— each with its own audit trail arcane and format—to securely monitor their systems and comply with requirements for log management and monitoring.

About the Author

Randy Franklin Smith is president of Monterey Technology Group, Inc. and creator of the UltimateWindowsSecurity.com Web site and training course series. As a Systems Security Certified Professional (SSCP), a Microsoft Most Valued Professional (MVP), and a Certified Information Systems Auditor (CISA), Randy specializes in Windows security. Randy is an award-winning author of almost 300 articles on Windows security issues for publications such as Windows IT Pro, for which he is a contributing editor and author of the popular Windows Security log series. He can be reached at rsmith@ultimatewindowssecurity.com.

About Quest Software, Inc.

Now more than ever, organizations need to work smart and improve efficiency. Quest Software creates and supports smart systems management products—helping our customers solve everyday IT challenges faster and easier. Visit www.quest.com for more information.

Contacting Quest Software

PHONE 800.306.9329 (United States and Canada)

If you are located outside North America, you can find your local office information on our Web site.

E-MAIL sales@quest.com

MAIL Quest Software, Inc.
World Headquarters
5 Polaris Way
Aliso Viejo, CA 92656
USA

WEB SITE www.quest.com

Contacting Quest Support

Quest Support is available to customers who have a trial version of a Quest product or who have purchased a commercial version and have a valid maintenance contract.

Quest Support provides around-the-clock coverage with SupportLink, our Web self-service. Visit SupportLink at <https://support.quest.com>.

SupportLink gives users of Quest Software products the ability to:

- Search Quest's online Knowledgebase
- Download the latest releases, documentation, and patches for Quest products
- Log support cases
- Manage existing support cases

View the Global Support Guide for a detailed explanation of support programs, online services, contact information, and policies and procedures.



5 Polaris Way, Aliso Viejo, CA 92656 | PHONE 800.306.9329 | WEB www.quest.com | E-MAIL sales@quest.com

If you are located outside North America, you can find your local office information on our Web site

© 2010 Quest Software, Inc.
ALL RIGHTS RESERVED.

Quest Software is a registered trademark of Quest Software, Inc. in the U.S.A. and/or other countries. All other trademarks and registered trademarks are property of their respective owners.
WPW-UnixLinuxInTrust-Smith-US-MJ-20091223