Version 1.02 – 7/2/02

SECURE AUDITING FOR LINUX

SOFTWARE REQUIREMENTS SPECIFICATION (SRS)

July 2002

Table of Contents

List of Figures

1. Computer Software Configuration Item – Auditing Package

1.1 *Background*

Presently the GIG (Global Information Grid) is the main tool for collecting, storing, processing, and disseminating information to the war fighters, policy makers and supporters of the U.S. government. Defense Advanced Research Projects Agency (DARPA) has accepted proposals and funded individual projects to demonstrate a capability of influencing the open source community in the area of Information Assurance. The intent is to discover the probability of enhancing an open source operating system that is secure enough to certify for use within the GIG. This project focuses on several aspects within one open source community.

Although there are several open source OS's (Operating Systems) in use today this project selected the Linux OS. Linux is a computer OS similar to the Unix OS. Linux was developed by Linus Torvalds in the early 1990's. In late 1991, Linus introduced Linux on the Internet under the GPL (GNU Public License).

The decision to license Linux under GNU created a different type of computer OS from those being sold by major distributors of OS software. The Linux OS kernel is open source software that is controlled through Torvalds, yet through the GNU license the source code is available to anyone. There are no license requirements preventing changes to the source code and redistributing it under a new company, but the GNU license requires that the changed software must also include source code when being distributed. This Open Source strategy allows any software developer access to source code in order to test the capabilities and shortfalls of the Linux OS. Torvalds accepts comments and recommendations to the source code from the public and he utilizes a team of developers to make decisions on what each official release will consist of.

The Linux OS is distributed as a kernel. The kernel of the OS is the central functional component. The kernel controls access to computer hardware resources, such as the hard drive, printers and floppy disk drive, as well as the man-machine interface (keyboard, screen, etc). Unlike other OS's, such as Microsoft Windows, the Linux kernel does not include a window manager or desktop manager. These are software packages that can be added to a Linux machine after installation of the kernel to afford ease of use. The Linux kernel correctly installed gives you command line interaction only. Several window manager and desktop manager packages available for Linux have also been developed under the GNU license.

Open source software frequently requires a higher level of computer programming knowledge. Companies such as Microsoft and Apple Computer have built a business on ease of computer use for the customer. Due to large business demands these computer software companies have included Information Assurance security within their own OS software packages. Linux on the other hand has never mandated such internal security. Companies that have taken this on, such as Hewlett Packard, have maintained the Linux kernel as is and developed proprietary software to be added to the kernel. This software is not under the GNU license and as of yet carries a large per license charge.

Throughout the 1990's Linux gained in popularity yet continued to fall short of the security requirements of the U. S. Government's Global Information Grid. In order for OS's to be utilized in a government-networking environment they must follow certain Common Criteria (CC) requirements. Common Criteria has many aspects to its security. One of the aspects is that the Trusted Computing Base (TCB) must be able to create, maintain, and protect from modification or unauthorized access or destruction, an audit trail of accesses to the objects it protects. In order to engage in a manageable segment this project concentrates on secure auditing for the Linux kernel.

Presently there is a major lack of secure, reliable, scaleable, policy-based system auditing within the Linux kernel, especially any that proposes to audit activities at the kernel level. Although the Linux kernel operates extremely efficiently, it does not monitor internal program calls to the kernel itself. Such calls would include user logins/logoffs, file accesses, file creation/deletion, privilege modifications, etc; in order to maintain a level of Common Criteria security, it is important to log all users of the system and what commands they execute within the OS. By automatically auditing these system calls to the kernel, the Linux OS would meet the Common Criteria requirements for auditing. These audit logs are important for system administrators to audit the activities of system users and for forensic personnel to track and recreate illegal activity if the system has been compromised. This project investigates which system calls require auditing in order to meet the Common Criteria standards while minimizing the performance impact of auditing on the system. Special care will be taken to produce an architecture that allows the audit system to be secure, reliable, and scalable and to the greatest extent possible, configurable.

Securing audit data is the chief requirement of the SAL program. The safekeeping and confirmation that data sent to the logs has not been modified is its highest priority. In essence the SAL program establishes a chain of custody for the log data. This is especially important to prove in a court of law that the information presented to convict a perpetrator for an illegal action is true and accurate. This is an important feature for the U.S. Government and businesses alike. Additionally, this project also investigates different ways of securing the audited data as soon as it is written to audit logs. The safekeeping of data will not be assigned to a single system, instead audit data will be gathered, encrypted locally and transmitted to a secure data repository. During transmission to a repository, audit data could potentially travel over untrusted networks, so encryption techniques will be used to prevent possible eavesdropping.

Producing a system that is both scalable and configurable is extremely important for the SAL program. To be used by the U.S. Government and commercial businesses alike, the SAL program will need to be configurable enough to fit into any established Security Policy. These policies can vary from the simple logging of bad passwords to capturing every keystroke and network packet that an instrumented system receives. Configurability will also be important when the established security policies call out different levels of Quality of Security Services (QoSS). Different levels of QoSS provide system administrators the ability to instrument systems differently. Systems with high security requirements will get a fully instrumented kernel while systems with low security requirements will receive partially instrumented kernels. To support these multiple levels of QoSS SAL will develop an application protocol framework that is connection oriented and asynchronous. A final aspect to be incorporated will be a log viewer which will allow a system administrator and/or forensics professional the ability to parse, view or print log information in order to simplify audit log data extraction. This will be a read only function, the audit logs themselves will not be written to during this process to maintain the security of the logs.

A project goal of the SAL program is to have these aspects incorporated into a future release of the Linux kernel by Linus Torvalds.

1.2 *Scope*

- **Purpose of the Work**

The purpose of the work is to develop an auditing package for Linux (kernel version 2.4.2 as of this writing) that is compliant with the U.S. Government's C2 standards as well as Common Criteria standards for security. SAL will also provide a mechanism for collecting log information such that it is admissible as evidence in a court of law.

- **Benefits**

This auditing package will help to meet the information assurance goals for the GIG. It will also be of immediate benefit to law enforcement in computer forensics. Additionally, by residing at the kernel level this audit package will allow for all processes to be monitored.

- **Scope**

The first goal will be to provide a C2 compliant kernel-level logging facility. Some work has already been done in this area (Auditd and Linux BSM), but a limited number of events are logged. Discussions with law enforcement will be conducted to determine the requirements that must be met to allow this data to be admissible as evidence in court. Additional discussions will determine the level of configurability that is necessary for acceptance by the system administrator community.

- **Objective**

To create a C2 compliant logging facility, providing assurance that the data cannot be modified once it is recorded. The source code for this package will be released under the terms and conditions of the GNU Public License.

Further SAL is not to degrade system performance by a significant margin on an instrumented system.

- **SAL System Design**

There are three system components in the SAL architecture: the Log Generator (LG), the Log Relayer (LR), and the Data Repository (DR).

- A machine that can generate an audit log message will be called a “generator”.
- A machine that can receive the message and forward it to another machine will be called a “relay”.
- A machine that received the message and does not relay it to any other machine will be called a “repository”. This will be known as the “SAL server.”
- Any generator or relay will be known as the “sender” when it sends a message.
- Any relay or repository will be known as the “receiver” when it receives the message.

The diagram below shows some of the components involved with a basic implementation of SAL. It depicts a “Linux-based PC with Auditing” generator as the sender of audit information to a “Central Logging Server” repository. There are many other implementations of SAL that may or may not include one or more relays and one or more repositories.

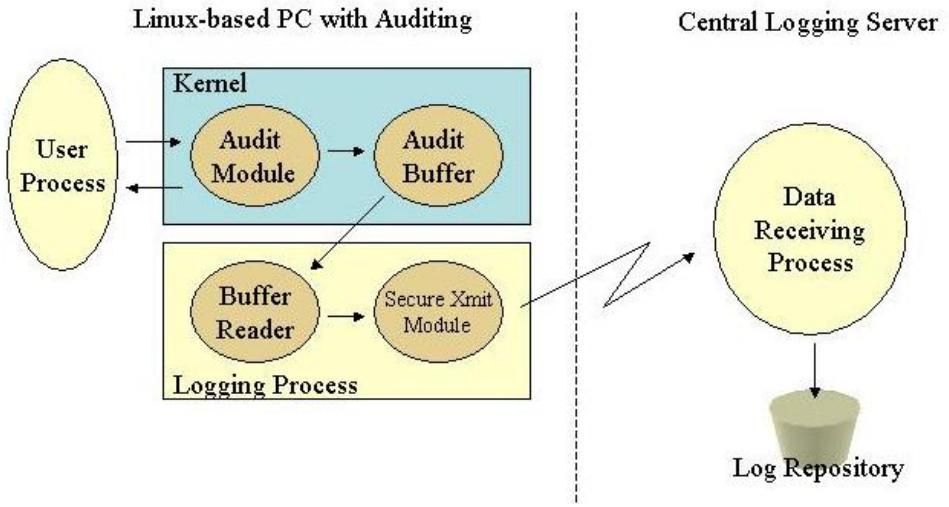

Figure 1 - SAL System Design

1.3 *Document Overview*

This document will identify the requirements for the Auditing Package Computer Software Configuration Item (CSCI).

This Requirement Specification describes what the Auditing Package CSCI is required to functionally perform, along with its required input and output. The Auditing Package will only function as specified within this document. For a definition of the terms, refer to the Glossary section of this document.

2. Requirements

The following requirements apply to the Auditing Package Computer Software Configuration Item (CSCI).

The overall requirement is to create an auditing package for the Linux operating system, kernel version 2.4.2, which is compliant with government "C2" standards for security. SAL will also provide a chain of custody mechanism that allows the collected and stored information to be admissible as evidence in a court of law.

2.1 *Functional Requirements*

The following section contains the functional requirements, organized by feature, that define the Auditing Package.

2.1.1 General Requirements

The following are goals and requirements that generally apply throughout the Secure Auditing for Linux (SAL) Package.

Req No	Requirement Description	Derived From	Common Criteria	C2
GEN010	The goal of SAL is to audit all events specified in section 2.1.2 during the period beginning from complete system startup and ending with the initial shutdown sequence.			
GEN020	SAL will protect audit data from unauthorized modification and deletion.	DoD 5200.28-STD 2.2.2.2, FAU_STG 1	✔	✔
GEN030	SAL will protect audit data from unauthorized viewing.	DoD 5200.28-STD 2.2.2.2, FAU_SAR 2.1	✔	✔
GEN040	The goal of SAL is to be included in the kernel as a static module.			
GEN050	The goal of SAL is to be configurable to fit into a variety of established security policies.			
GEN060	SAL will define three security policies. One of the three will meet the level of C2 auditing.			
GEN070	The goal of SAL is to scale from a single computer with an instrumented kernel and an external data repository to supporting many hundreds of instrumented computers with multiple data repositories.			

GEN080	SAL will provide a verifiable chain of custody record for all audit log data from the moment it is created on a generator to when it is stored inside a data repository.			

2.1.2 Recording Requirements

The goal of SAL is to be capable of recording any of the following events in the context of kernel system calls. Some events map one-to-one with particular system calls while other events map to multiple system calls. Each event configured for auditing will be recorded and sent to a repository.

Req No	Requirement Description	Derived From	Common Criteria	C2
REC010	SAL will audit logins and logoffs.	DoD 5200.28-STD 2.2.2.2		✔
REC020	SAL will audit process invocation and terminations.	DoD 5200.28-STD 2.2.2.2		✔
REC030	SAL will audit file or file system accesses and modifications.	DoD 5200.28-STD 2.2.2.2		✔
REC040	SAL will audit security privilege changes.	DoD 5200.28-STD 2.2.2.2		✔
REC050	SAL will audit user account modifications.	DoD 5200.28-STD 2.2.2.2		✔
REC060	A goal of SAL is to audit Inter-process communication command invocations.			
REC070	A goal of SAL is to audit network configuration modifications.			
REC080	A goal of SAL is to audit memory access command invocations.			
REC090	A goal of SAL is to audit hardware device configuration modifications.			
REC100	SAL will audit startup and shutdown of audit logging processes.			
REC110	A goal of SAL it to audit run-time kernel configuration modifications.			
REC120	SAL will audit retrieval of system-related information.			
REC130	SAL will audit directory operations and traversals.			

REC140	SAL will audit system configuration modifications exclusive to superuser.			
REC170	Audit data will be stored in an Audit Repository.			
REC180	Audit data that is stored locally will be tagged as such when being stored at an audit repository so that proper knowledge of such is logged			
REC190	Audit data will be stored such that chain of custody will be maintained and verifiable.			
REC200	All changes to SAL's configuration will be audited.			

2.1.3 Reporting Requirements

SAL will be designed to accept messages from other logging systems, such as syslog. While there is no format imposed on these messages, SAL generated messages will contain specific information for each event.

Req No	Requirement Description	Derived From	Common Criteria	C2
REP010	No assumption is made about the formatting or content of the message being logged at a repository.			
REP020	SAL will accept and log any audit messages it receives from authenticated generators.			
REP030	SAL will support the concept of Facility (Similar to syslog)			
REP040	SAL will support the concept of Severity (Similar to syslog)			
REP050	Valid SAL messages will contain a message header, which will have a representation of date and time of the event in UNIX time format (number of seconds since January 1, 1970) including a 4 digit year.	DoD 5200.28-STD 2.2.2.2, FAU_GEN 1.2	✔	✔
REP060	Valid SAL message headers will contain the identification of the sending generator.			
REP070	Valid SAL message will contain a message payload.			
REP080	For kernel messages a numerical system call identifier(s) associated with the type of event will be included.	DoD 5200.28-STD 2.2.2.2, FAU_GEN 1.2	✔	✔
REP090	For kernel messages a numerical representation of system call return status indicating the success or failure of the event will be included.	DoD 5200.28-STD 2.2.2.2, FAU_GEN 1.2	✔	✔

REP100	For kernel messages a numerical identification of the user initiating the event will be included.	DoD 5200.28-STD 2.2.2.2, FAU_GEN 2.1	✔	✔
REP110	For kernel messages a numerical identification of the terminal originating the event will be included.	DoD 5200.28-STD 2.2.2.2		✔
REP120	For kernel messages a numerical and text identification of the process spawning the event will be included.			

2.1.4 Encryption Requirements

Req No	Requirement Description	Derived From	Common Criteria	C2
ENC010	The goal of SAL is to encrypt audit data in the kernel.	FCS_COP 1.1	✔	
ENC020	The goal of SAL is to securely generate and manage kernel encryption keys of at least 128 bits in length.	FCS_CKM.1.1	✔	
ENC030	The goal of SAL is to securely transmit the audit data and encrypted key to the log server at regular intervals			
ENC040	The goal of SAL is to include a verifiable signature with each transmission of audit data to the log repository.			

2.1.5 Performance Requirements

Req No	Requirement Description	Derived From	Common Criteria	C2
PRF010	When fully operational, not in a faulted or error state, and the network is available and operating normally, the goal of SAL is to deliver audit data to the log repository in at most three minutes - measured from when the event occurs to when the log entry and key reach the repository .			
PRF020	The goal of SAL is to store audit data at Audit repositories. In the event that transmission to a repository is currently unavailable, local logging will be initiated that will be flagged as suspect when received by the repository.			

PRF030	Whether stored locally or on the remote server, the goal of SAL is to store audit data according to time (in UNIX time format).			
PRF040	The goal of SAL is to be free of buffer-overflow issues and other known attacks, including denial of service attacks.			

2.1.6 Systems Administrator Utility Requirements

The following requirements apply to the utilities provided for system administrators.

Req No	Requirement Description	Derived From	Common Criteria	C2
SAU010	The goal of SAL is to provide the ability to generate and maintain encryption keys as appropriate for secure transmission of audit data to a log repository.	FCS_CKM 1, FCS_CKM 2	✔	
SAU020	The goal of SAL is to provide the ability to select audited events based on one or more criteria including user identity.	DoD 5200.28-STD 2.2.2.2, FAU_SEL 1.1	✔	✔
SAU030	The goal of SAL is to provide the ability to query audit data for log entries pertaining to a user id, time range, system call identifier, or process name.	FAU_SAR 3.1	✔	
SAU040	The goal of SAL is to provide the ability to analyze audit data.	FAU_SAA 4	✔	
SAU050	The goal of SAL is to provide the ability to export audit data in text format for the purpose of printing.			
SAU060	The goal of SAL is that only tools available in the Open Source Community be used for its configuration, installation and maintenance once installed.			
SAU070	Configuration of SAL will be accomplished by a configuration tool. This tool will walk an administrator through the steps required to implement SAL in a system.			
SAU080	The configuration tool will support a minimum of 4 levels of QoSS (HARDENED, MED, LOW, USER DEFINED). Policies will be defined for HARDENED, MED and LOW levels of QoSS.			
PRF050	SAL will allow the configuration of the audit server by way of a configuration tool.			

2.1.7 Standards Compliance

In an effort collaborate with and support the Open Source community the SAL program will use existing standards. In the event that the existing standard fall short of our requirements they will be extended. The extensions of any RFC will be brought back to the community for review. The specific RFCs are:

RFC3164 - BSD syslog protocol (the old)

RFC3080 - BEEP, protocol used to transmit syslog-reliable messages

RFC3081 - BEEP over TCP

Req No	Requirement Description	Derived From	Common Criteria	C2
STD010	SAL will develop a Policy-Based Log Auditing Data Storage format that will be accepted in a court of Law.			
STD020	SAL will implement a Policy-Based Log Auditing Data Storage format.			
STD030	A goal of SAL is to be compatible as a repository with the protocols defined in RFC3164, RFC3080 and RFC3081.			

2.2 *External Interface Identification*

This section lists all external interfaces, both input and output, and identifies which interfaces have fixed interface characteristics (and therefore impose interface requirements on the Auditing Package), and which interfaces are being developed or modified (thus having interface requirements imposed upon them by the Auditing Package).

2.3 *External Input Interfaces*

This section identifies the external input interfaces and the input data required by System Design.

2.3.1 User Interface

The following requirements define the user external input interface and the entries that SAL accepts from a user:

Req No	Requirement Description	Derived From	Common Criteria	C2
USR010	SAL shall provide a command line interface, with a			

	goal of using a graphical interface, for any configurable options on the data repositories.			
USR020	The goal of SAL is to provide a command line interface for any configurable options on the audited system.			
USR030	The goal of SAL is to support the use of a configuration file to configure all aspects of the program (generator, relay, repository).			

2.4 *External Output Interfaces*

This section identifies the external output interfaces of System Design, including the output data System Design will be required to produce, for each external interface.

2.4.1 User Interface

The following requirements define the user external output interfaces and identify the output that SAL has to the user.

Req No	Requirement Description	Derived From	Common Criteria	C2
USR040	SAL shall provide a command line user interface, with the goal of using a graphical interface, for the reporting and analysis of audit data on the log repositories.			
USR050	SAL viewing tools will have the ability to display as well as print all selectable views.			
USR060	SAL shall provide filtering capabilities to any tool used to view log data.			

2.4.2 Internal Interfaces

The following requirements define the internal output interfaces.

Req No	Requirement Description	Derived From	Common Criteria	C2
INI010	To remove the need for administrators to support multiple loggers, a goal of SAL is to interface with the standard Linux syslog program. SAL will act in this regards as a Syslog Collector, so that audit data can be stored to a protected secure data repository from non-instrumented systems.			

2.4.3 Computer Resources

Req No	Requirement Description	Derived From	Common Criteria	C2
CMP010	The goal of SAL is to work with the most current commercial release of the Red Hat distribution of Linux.			
CMP020	The goal of SAL is to work on a minimum of an x86-based Pentium 233 MMX or AMD K6-233 with 64 megabytes of RAM.			
CMP030	SAL will be able to function as a headless system at the data repositories.			
CMP040	SAL's installation disk space requirement goal is to not consume an amount of disk space greater than ten megabytes including source code, compiled code, and all necessary third-party components.			
CMP050	A goal of SAL is to support large data repositories (greater than 10 Gigabytes).			
CMP060	SAL's data repository will be capable of using additional storage space if it becomes available after repository startup.			

3. Constraints

This section identifies constraints imposed on the Auditing Package:

3.1 *Third-party Software*

If the installation of any utilities or libraries are required, these will be clearly indicated and discussed in any install documentation. Any such software will not restrict the open source nature of the Auditing Package.

3.2 *Audit Data Log Size*

The size of the audit data log is restricted to the amount of free disk drive space available to the data repository. However SAL will be developed such that there will be no maximum disk storage limitation, and furthermore will be able to use additional storage resources if they become available during execution.

3.3 *System Clock*

In order to ensure accuracy of the audit data timestamps, the system clock on the audited system and log server (if applicable) must be set as accurately as possible and periodically adjusted if necessary. When audit data is generated a time stamp is used. This time stamp is maintained with the audit data throughout its existence. When a repository receives audit data the time it was received will be logged and will also stay with the data through out its existence.

4. Validation Criteria

A Software Test Plan (STP) will be created to define a set of qualification methods and to specify for each requirement in this document a method of ensuring that the requirement is satisfied.

5. Documentation

The following documentation will be developed and maintained to support this project and the end users:

- System Requirements Specification : This document provides the requirements and sources for the system. It also describes the system from a high level, without providing any of the design.
- Software Project Management Plan : This document establishes the development process for the project.
- Software Design Document : This document presents a high level design for the system, decompose this into components, and describe each component.
- Software Test Plan : This document describes how the requirements will be tested in the final system.
- User's Guide : This document describes the use of the system for day-to-day activities. It is intended as a reference for the end user, to supplement any on-line or built-in help.
- Installation Guide : This document details the installation and initial configuration of the system, to include selection of the appropriate policies.

6. Bibliography

This section contains a master list of all documents and other sources of information referenced in this CSCI SRS:

Department of Defense Trusted Computer System Evaluation Criteria, Department of Defense, December 26, 1985.

Evaluation Criteria for IT Security, ISO/IEC 15408-2, December 1, 1999.

RFC3164, "The BSD syslog Protocol", C. Lonvick, Copyright (C) The Internet Society (2001)

RFC3080, "The Blocks Extensible Exchange Protocol Core", M. Rose, Copyright (C) The Internet Society (2001)

RFC3081, "Mapping the BEEP Core onto TCP", M. Rose, Copyright (C) The Internet Society (2001)

7. Glossary

This section contains a list of project definitions and acronyms used throughout this document:

API	Application Programming Interface
BEEP	Blocks Extensible Exchange Protocol
BSD	Berkley Standard Distribution
C2	Class C2 is a rating granted by the National Computer Security Center (NCSC) for products that have been evaluated against the Department of Defense Trusted Computer System Evaluation Criteria (TCSEC).
CC	Common Criteria
COTS	Commercial-Off-The-Shelf
CSCI	Computer Software Configuration Item
DARPA	Defense Advance Research Projects Agency
Data Repository	Repository
DoD	Department of Defense
Generator	SAL implementation that sends audit messages
GIG	Global Information Grid
GPL	GNU Public License
GUI	Graphical User Interface
ID	Identification
Log Generator	Generator
Log Relayer	Relay
OS	Operating System
PC	Personal computer
QoSS	Quality of Security Service
Receiver	Any relay or repository will be known as the receiver when it receives a message
Relay	SAL implementation that relays audit messages to a SAL repository
Repository	SAL implementation that stores audit data to a secure recording device.
SAL	Secure Auditing for Linux

Sender	Any generator or relay will be known as the sender when it sends a message
SPMP	Software Project Management Plan
SRS	Software Requirements Specification
SSS	System/Subsystem Specification
STP	Software Test Plan
Syslog	Standard Linux application that currently logs data to a repository. (Not secure or reliable)
TCB	Trusted Computing Base
TCP	Transmission control protocol

8. Issues

There are no issues at this time.