

Constructing a High Assurance Mail Guard

Secure Computing Reprint Series
Richard E. Smith
1994

SECURE
COMPUTING

**Secure Computing Corporation
Corporate Headquarters**

One Almaden Boulevard, Suite 400
San Jose, CA 95113-2211

tel +1.800.379.4944

tel +1.408.918.6100

fax +1.408.918.6101

European Headquarters

tel +44.1753.826000

fax +44.1753.826001

Asia/Pacific Headquarters

tel +65.535.6206

fax +65.535.8138

www.securecomputing.com

T A B L E O F C O N T E N T S

CONSTRUCTING A HIGH ASSURANCE MAIL GUARD

Abstract3
Introduction3
Mail guard structure5
Why high assurance6
Trusted software development7
SMG formal assurance7
Performance7
Acknowledgments9
References9



Abstract

This paper describes the mail guard constructed as part of the Secure Network Server (SNS) Development Program.

The SNS Mail Guard (SMG) provides a highly trustworthy device for transferring electronic mail between networks of differing security levels in accordance with site specific policies. The site defines its message transfer policies based on specific tests of message contents. The development effort pursued high assurance through compliance with trusted software development requirements and through formal assurance of security properties. The resulting mail guard uses the type enforcement capabilities of the LOCK® trusted computing base (TCB) to provide the most trustworthy facility achievable with current technology. We have found that high assurance security does not visibly impact mail guard performance.

Introduction

The Secure Network Server (SNS) Development Program applies the LOCK® Trusted Computing Base (TCB) [6] to network security services. The SNS program's goal is to provide a set of useful networking facilities that achieve high security assurance [7].

The first phase of SNS has produced the SNS Mail Guard (SMG), a device capable of controlled reclassification of electronic mail (e-mail). The SMG connects to local networks that use the Internet protocol suite and the Simple Mail Transfer Protocol (SMTP). Users on such networks operating at different security levels can use the SMG to exchange e-mail in a controlled fashion (Figure 1).

Organizations that handle sensitive or classified data generally establish separate computer networks for each sensitivity level of data they must handle.

Each network operates in a "System High" mode without security labels to indicate the sensitivity of its data.

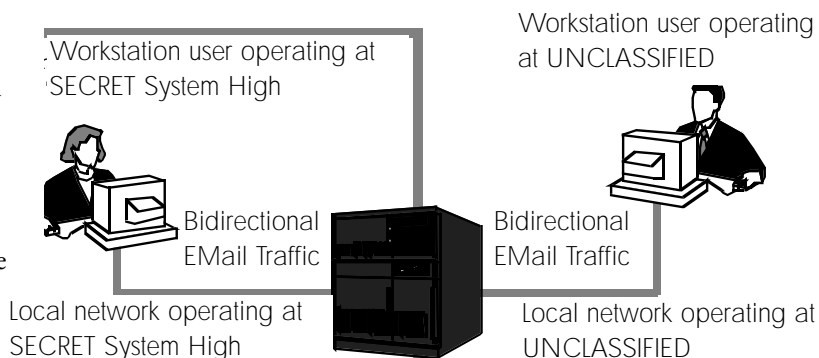


Figure 1: The SMG allows users inside a protected, System High enclave to communicate via Unclassified electronic mail with users outside the enclave.

This is because cost effective, commercially available equipment never provides security labels reliable enough for such applications. By default all data in such networks is implicitly labelled according to the most sensitive data thereon.

E-mail connectivity has become so important that disconnected groups and organizations suffer a recognized operational disadvantage. Today, this is the common fate of groups operating on a classified network. Commercial equipment is not built to keep classified information separate from unclassified. If a classified user composes an unclassified message, there must be a special facility to reliably release the unclassified information to the unclassified network.

This facility must be highly trustworthy to prevent the wrong information from flowing between the networks. This is the purpose of the high assurance SMG.

The SMG accepts e-mail messages from one network and, according to the destination address, routes them through a reclassification procedure (Figure 2).



If the procedure approves the message for reclassification, then the message is reclassified and passed to the appropriate network for delivery. The decision making process for reclassification is implemented using one or more special procedures called “filters.”

The choice of filters is controlled according to site specific policy decisions and configured by the

SMG’s site administration. Different filters may be applied to e-mail traffic depending on the e-mail’s source and destination networks.

The architecture of the SMG allows the integration of a variety of filters, depending on the release requirements for the site using the guard. Individuals composing e-mail on personal workstations protected behind an SMG must ensure that the message contents conform to the site’s release requirements. Some filters may require that individuals use special software or hardware (like cryptographic services) at their workstations.

The following describes several filters being produced by the SMG program. Filters are individually enabled or disabled according to site security requirements. Each makes its message release decision based on detecting specific types of information in an e-mail message submitted for reclassification:

- SMTP sender or recipient addresses. The filter compares the name of the message’s sender and its recipients against a database of addressees. The sender and recipients must all be allowed to send or receive e-mail through the SMG.
- Classification label. The filter searches the body of a message for a line of text indicating the security classification of the message’s contents. The author of the message must insert the label into the message to specify the sensitivity of the message’s contents.
- Attachment file types. The filter searches the message for attached files in a variety of application

specific formats. Each attached file must be of a type that is permitted to traverse the SMG. A site can use this facility to block the accidental importation of executable binary files that may contain virus software.

- Attachment review indicator. The filter searches attached files for a special tag and checksum to indicate that the file had been reviewed by special software (the “attachment review module” or ARM) on the sender’s personal workstation. If the site requires attachment review, then the SMG will transmit the message only if attachments it contains have been reviewed using the ARM.
- MOSAIC/MSP digital signature. The filter verifies that the body of the message is formatted according to the Message Security Protocol (MSP) and signed using the MOSAIC digital signature algorithm [3]. The SMG will transmit the message only if the message is signed with a valid signature. The filter may also verify that the signature certificate belongs to an individual authorized to send e-mail through the SMG.
- MOSAIC/MSP encryption. The filter verifies that the body of the message is formatted according to the Message Security Protocol (MSP) and the message text has been encrypted and signed using MOSAIC. The SMG will transmit the message only if its contents are properly encrypted.

If a site decides to allow e-mail to flow in a given direction between two networks, the site must choose which filters will be applied to that e-mail traffic. The choices must maximize the likelihood that reclassification and release decisions are based on information produced by the witting act of an authorized individual rather than on accidental or corrupted contents of an e-mail message. The SMG will typically base its reclassification decisions on information produced or transported by unassured commercial equipment, since that is the equipment in common use today.

SECURE
COMPUTING

The choice of filters, then, depends on the security properties of the networks being connected to the SMG. Small, isolated local networks might use physical security and strong configuration controls to ensure the integrity of e-mail passed to an SMG. Larger, less controlled networks may require the stronger evidence of user identification and message integrity provided by a cryptographically protected digital signature.

Mail guard structure

The SMG combines off the shelf networking software with specially developed guard software, hosting both on the LOCK TCB. A common problem in such systems is to keep the less trustworthy off the shelf software separate from the more trustworthy guard software. It is important to ensure that flaws in the off the shelf software will not prevent the guard software from doing its job. The underlying TCB must protect the integrity of the different software components from one another, and it

must ensure that the guard software is never bypassed.

On LOCK, we rely on the Type Enforcement™ facility to achieve this. Type Enforcement technology is a special form of mandatory, rule based access control provided in addition to conventional, label based access control rules. Like access control based on labels, Type Enforcement technology completely prevents a program from reading or writing data items unless it is specifically allowed to under the system's security policy. Unlike label based access control, type enforcement is associated with particular programs and particular types of data files.

By placing type restrictions on collections of programs and data items, we can require data to flow through a group of programs in a specific order. This allows us to take data from a program of dubious integrity, pass the data through another program that “censors” it or takes other measures to insure the data's integrity, and then pass it to a third

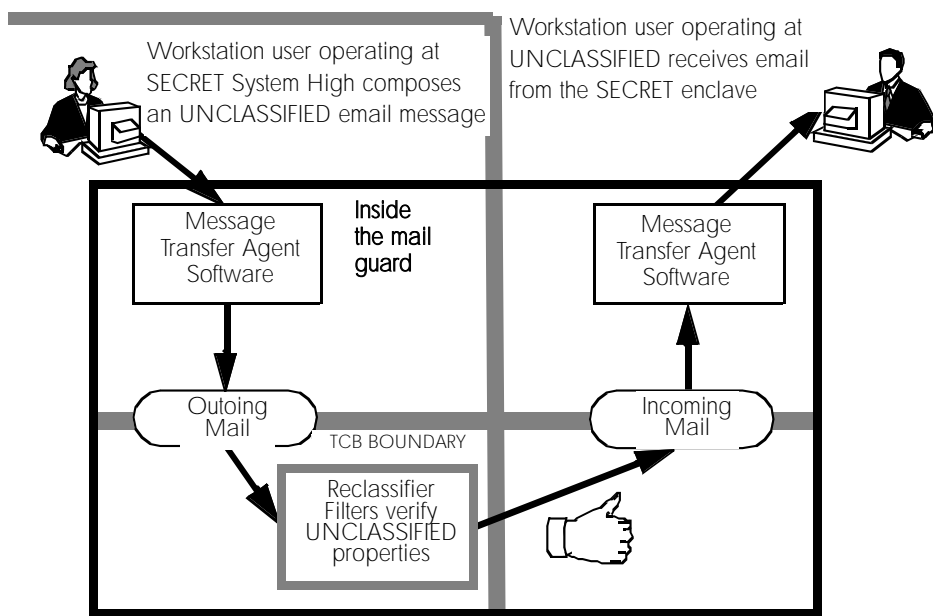


Figure 2: The SMG uses LOCK's type enforcement to isolate the behavior of its off the shelf message transfer agent software from its reclassification procedures.



program that assumes the data has the established integrity properties. This technique is called an “assured pipeline” and is fully described in [4].

The SMG implements an assured pipeline to allow its mixture of software components to interact effectively while preserving necessary security properties. Figure 2 illustrates the principal components of the mail guard:

- Message transfer agent (MTA) software
- Reclassifier software (gray bordered box)
- Incoming and outgoing mail queues (ovals)
- Boundary between security levels (dashed line)
- TCB boundary (gray line)

None of the programs illustrated here run in any form of kernel or “root” mode with unlimited access privileges. All these programs are constrained by type enforcement so that they operate as an assured pipeline.

The mail guard pipeline has a message transfer agent at each end and the reclassifier in the middle. The reclassifier itself consists of separate programs in separate domains to enqueue messages for reclassification, invoke the appropriate filters, and to write the approved messages to the incoming queue for the receiving message transfer agent.

The message transfer agents are not allowed to read messages across the boundary between security levels in either direction. This forces all reclassification to go through filters, where virus checks on incoming executable files and other such activities may occur.

Why high assurance?

The purpose of high assurance in the SMG is to ensure that reclassification tests are always performed and never bypassed, regardless of how strong or weak the tests themselves might be. This provides command authorities at the site with cer-

tainty that reclassification and release decisions are made in accordance with the specified policy.

Furthermore, the authorities must be certain that they can reliably modify the SMG’s policy to respond to changes in the network configuration and perceived threat. High assurance techniques in trusted software development increase the visible trustworthiness of the resulting system. Procedural requirements for trusted software development assure that all developmental steps are carefully thought out and the results are consistently checked. Formal models and analyses of the system’s architecture increase the likelihood that all security relevant design flaws have been eliminated. In our experience on the LOCK development program, analysis also uncovers subtle features, design constraints, and interdependencies that would not otherwise have been recognized. This leads to a more thorough understanding of the overall system, its security requirements, and its limitations.

A mail guard is essentially a filter that restricts the flow of data so that “acceptable” messages flow from one security level to another, and “unacceptable” messages do not. Physical security provides assurance that data only flows between the levels via the guard. Walls, floors, protected cabling, and physical access control measures prevent data from flowing between levels except through the guard. The LOCK TCB provides a solid framework which prevents data from flowing between levels except when passing through the filter software. The LOCK formal assurance work provides the detailed look at the trustworthiness and effectiveness of that framework. The mail filters are made modular with respect to the LOCK formal assurance; they are developed and tested separately, and their behavior is analyzed as an extension to the existing system.

It is important to note that formal assurance does not provide some unilateral stamp of quality. All that assurance can do is indicate that the system preserves some specific property. In the case of the

SECURE
COMPUTING

SMG, the formal assurance focuses on preservation of label based, mandatory access control on data handled by the SMG.

Trusted software development

The SMG development contract mandated that all TCB software be developed in compliance with trust requirements of the Trusted Software Methodology (TSM) developed by the Strategic Defense Initiative Organization [2]. The specific requirements called out in the contract were roughly similar to those associated with Level 3 of the Software Engineering Institute's Capability Maturity Model [5]. The TSM rates software development trustworthiness on a scale from T0 ("untrusted") to T5 ("highly trustworthy"). The SMG program specifies a mixture of requirements ranging from T3 to T5. The T3 requirements primarily apply to environmental and organizational policy issues: properties of the software development environment, for example. The T5 requirements are associated with software development procedures: design analysis, test case development, and formal reviews, for example.

Both contractor and government personnel responsible for the SMG program were trained in the TSM. This training provided the rationale for the requirements and an overview of how they might be applied. The training course made it clear that perfect compliance with the TSM was in fact beyond the state of the art in software engineering. The recommended approach to this problem is to document all requirements that could not be complied with and explain how the program will handle the associated risks. The SMG program inherited many of its development tools and procedures from the LOCK program, which started in the mid 1980s. This placed practical constraints on the feasibility of incorporating radically new tools or procedures for complying with TSM requirements.

An internal review found that it was not feasible to comply with the letter of approximately 8% of the 438 TSM requirements that applied to the program. These requirements were analyzed with respect to risk to the program, documented, and submitted to the contracting organization for subsequent approval.

An important difference between SMG development philosophy and that of the TSM is the selective application of formal assurance. The TSM implicitly focuses on service assurance: the developed systems must provide faultlessly reliable service and not be sensitive to internal or external denial of service attacks. SMG requirements, on the other hand, focus on the highest possible assurance of security properties. High assurance was not applied to portions of the system whose effects on security could be otherwise constrained. The SMG incorporates "off the shelf" components (hardware and software) and uses a variety of hardware and software mechanisms to constrain their effect on the system's security properties.

The TSM requirements mandated by the contract are applied to all TCB software. This does not include "off the shelf" software, so the TSM was not applied to the TCP/IP software or the message transfer agent (MTA) software. These are constrained by high assurance hardware and software mechanisms so that they can not directly interfere with system security. All software that makes security policy decisions is subjected to formal assurance as well as the TSM requirements.

SMG formal assurance

Formal assurance techniques provide the basic evidence that a system is fundamentally trustworthy. It provides a distinctively global view of the system's security properties that can not be duplicated through postdevelopment inspection or test. It also enforces a special discipline on the system development process that focuses early attention on potential security problems. The analytical products of

formal assurance provide strong arguments for the soundness of a TCB design, profoundly increasing the TCB's trustworthiness.

SMG assurance focuses on assuring label based access control. The assurance shows that a message will be reclassified if and only if the message presented to the reclassifier is found to be acceptable by whatever filters are applied to it. The assurance-work says nothing about the behavior of entities outside the TCB, which are primarily off the shelf Internet protocol components. Nor does the assurance work guarantee availability or performance. Since the assurance focuses on label based access control, it does not address enforcement of need to know.

The challenge in TCB development is to extend the TCB's capabilities without invalidating previous assurance work. Using LOCK, we proceed by showing that the SMG enhancements either preserve or are constrained by LOCK's existing security mechanisms. The enhancements consist of network device drivers, the TCP/IP protocol stacks, and the MTA software.

Device drivers on LOCK execute in user mode. Each interface is assigned a single security label, so the driver is unable to affect label based access control decisions. Thus, the network device drivers are constrained by TCB access enforcement mechanisms.

The same approach is applied to the TCP/IP protocol stacks. Separate instances are provided for each distinct security label and each is tasked with processing network traffic at its single security level.

The same holds true for the message transfer agent software. Therefore, all off the shelf protocol software is constrained by the TCB and unable to influence label based access control decisions. This limits the amount of analysis required to show that the message transfer agent software maintains label based access control with high assurance. As discussed previously, authentication and identity based access control for e-mail are not handled by the LOCK TCB. The level of assurance required is subject to site policy depending on the facilities available on the networks served by the SMG.

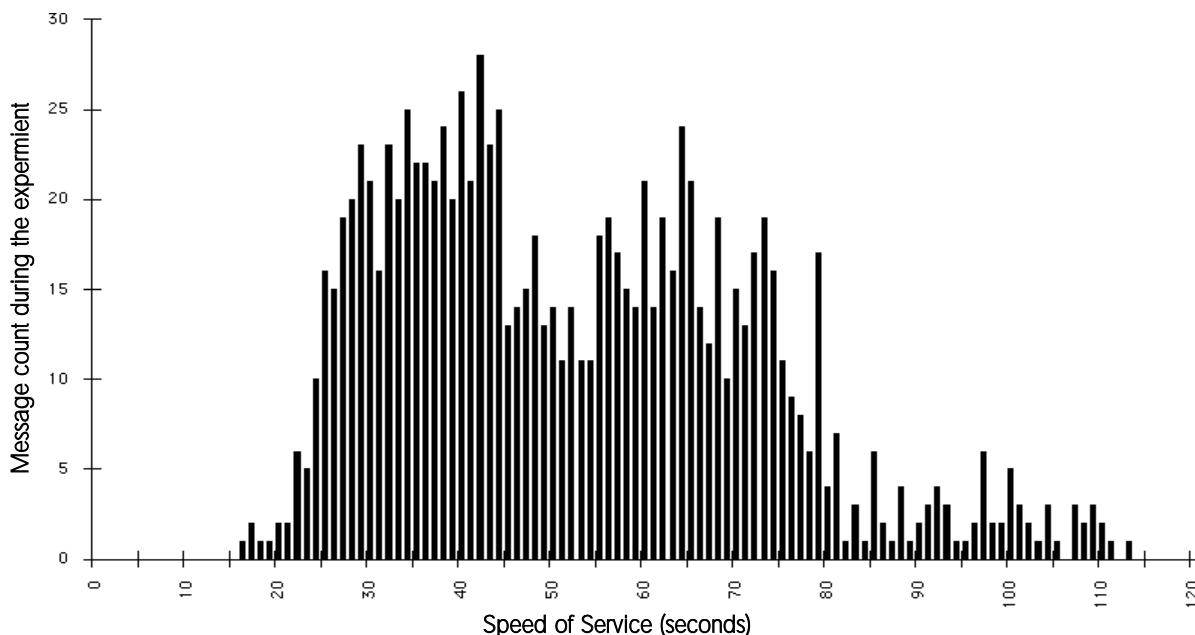


Figure 3: In performance tests, the SMG achieved an average speed of service of 52 seconds (the time to send a message from one host through the guard to another host, with null security filters). Security filtering adds an average worst case overhead of 53%.



Performance

An astonishing result of the original LOCK development effort was its level of performance. Standard benchmarks had trouble measuring a consistent difference between the performance of LOCK/ix, LOCK's Unix compatible environment, and that of an unmodified commercial Unix system operating on similar hardware. Uncertainties in benchmarking techniques make all such measurements suspect, but the results make it clear that high security assurance has not significantly reduced LOCK's performance. Measurements of the SMG exhibit similar performance.

Measurements of the MTA software running on the SMG show negligible difference in performance against the same software running on a comparable commercial Unix platform. MTA efficiency is particularly important in the SMG since each reclassified message must pass through two separate MTAs. An independent test and evaluation contractor found that the SMG consistently kept up with commercial mail server software.

Figure 3 shows the results of performance tests on the SMG. These tests measured speed of service while handling bidirectional mail traffic consisting of 10,000 byte messages. The tests did not use security filters. The analysis found the average delivery time ("speed of service") was 52 seconds for messages transmitted across the SMG, with a worst case of 113 seconds.

Filter	Speed of Service (seconds)
No Filtering	52
Access Control Filtering	75
Digital Signature (estimated)	76
Encryption (estimated)	80

The above table compares the SMG's average speed of service when applying different types of security filtering. The first two entries are based on direct

measurement of SMG performance. The second two are estimates based on an SMG performance model incorporating the measured performance of the Tessera crypto card. These figures are all within the speed of service requirements identified for the Defense Message System, providing an order of magnitude or better design margin for incorporating security filters on non-critical message delivery [1].

Acknowledgments

This paper reports the efforts the very capable SMG development team of Secure Computing Corporation's Sever Products Group. The formal assurance work benefited from the valuable contributions of Secure Computing's research organization.

This work was supported by Contract MDA904-93-C-C034. The author also thanks the anonymous reviewers for their constructive comments.

References

- [1] DoD, "Defense Message System (DMS) Required Operational Messaging Characteristics (ROMC)," Department of Defense, 23 April 1993.
- [2] GE Aerospace, "Trusted Software Methodology Volume 1: Trusted Software Program Demonstration, Assessment and Refinement," SDI-S-SD-91-000007, Strategic Defense Initiative Organization, Washington DC, 17 June 1992.
- [3] MOSAIC Program Office, "MOSAIC Program Overview," Version 2, Ft. Meade, MD, 28 January 1994.
- [4] Richard O'Brien and Clyde Rogers, "Developing Applications on LOCK," Proceedings of the 14th National Computer Security Conference, Washington, DC, October 1991, page 147.



[5] Mark C. Paulk, Bill Curtis, Mary Beth Chrissis, and Charles V. Weber, "Capability Maturity Model for Software," Report CMU/SEI-93-TR-24, Software Engineering Institute, Carnegie-Melon University, Pittsburgh, PA, February 1993.

[6] O. Sami Saydjari, Joseph M. Beckman, and Jeffrey R. Leaman, "LOCK Trek: Navigating Uncharted Space," Proceedings of the IEEE Symposium on Security and Privacy, Oakland CA, May 1989, page 167.

[7] Richard Smith, "MLS File Service for Network Data Sharing," Proceedings of the Ninth Annual Computer Security Applications Symposium, Orlando FL, December 1993.