# Analysis and Assessment of SELinux and McAfee HIPS

Edward Sealing, CISSP, RHCSA  
*Sealing Technologies Inc.*ed.sealing@sealingtech.org

# McAfee HIPS 8.0 for Linux - Kernel Modules

The McAfee Host Intrusion Prevention System (HIPS) 8.0 contains two kernel modules for policy verification and utilizes the Linux Security Modules (LSM) framework for policy enforcement. This following lists the modules and purpose.

1. **hipsec.ko** - Linux Security Module (LSM) Hook Driver. Sends events to user space policy for verification and waits for decision to block or allow access.
2. **schook.ko** - SysCall Hook. Makes a copy of the system call table in memory and hooks 29 of the kernel system calls.

Both kernel modules are licensed under the open source GNU Public License (GPL). The source code is provided with the HIPS installation package and is readily available for review.

# SELinux and HBSS HIPS Interoperability

The Linux security model used in all major Linux distributions can be broken into Discretionary Access Control (DAC) and Mandatory Access Control (MAC). The frameworks are implemented separately within the Linux Kernel. DAC is built into all Linux and provides a way for users to control access to owned files, and grant access where desired. MAC is implemented is not implemented directly in the Linux Kernel. Rather a modular architecture is provided in which security frameworks can be plugged into the Linux Kernel and implement MAC security. This framework is known as the the Linux Security Modules (LSM). Figure 1 shows how LSM is implemented within the Linux Kernel. There are multiple open-source modules that are currently accepted in LSM: SELinux (Red Hat and derivatives), Apparmor (Ubuntu Linux), Smack, and TOMOYO. McAfee also implements their Host Intrusion Prevention System (HIPS) as a MAC module that utilizes the LSM framework to provide policy enforcement.



Figure 1. Linux Security Architecture

Currently, LSM will only accept one module. Once a security module is registered, if any other modules attempt to register with the kernel, an error will be returned. For this reason, McAfee requires that SELinux be disabled prior to installing HIPS version 7.1 or above. Within the current Linux architecture, it is not possible to enforce both SELinux and HIPS policy together due to architectural limitations of the Linux kernel.

# Stacking multiple security modules simultaneously

According to the Linux Kernel Archives, it is theoretically possible to run multiple security frameworks at the same time, however LSM is not specifically designed to manage multiple modules. According to the Kernel Archives:

*"LSM also provides a simple mechanism for stacking additional security modules with the primary security module." "A security module can call these functions in order to stack with other modules. However, the actual details of how this stacking is handled are deferred to the module, which can implement these hooks in any way it wishes (including always returning an error if it does not wish to support stacking). In this manner, LSM again defers the problem of composition to the module."*

In a paper by Markus Quaritsch and Thomas Winkler, a concept was developed to stack multiple security modules by using a primary "multiplexor module" and having it maintain the security functions required for additional, unrelated modules to perform adequately. Additionally there is a project from 2006 on SourceForge called "Linux LSM Stacking" that provides patches to the Linux kernel that allow for stacking LSM modules, however these are not supported by Red Hat.

# Analysis of SELinux Security Services

SELinux was originally developed by the National Security Agency (NSA) as a MAC security architecture to be used in national security systems, and other platforms requiring a high level of trust. It was integrated into the mainline Linux Kernel with version 2.6.0 in 2003, and has been supported by Red Hat Linux, as well as many other vendors ever since.

SELinux can be placed in 3 modes of operation: Enforcing, Permissive, and Disabled. SELinux is enabled and in "Enforcing" mode by default on all currently supported versions (versions 4, 5, &6) of Red Hat Enterprise Linux (RHEL). The default policy shipped and enabled on RHEL is the "Targeted Policy" and is maintained by the Red Hat security engineering team. It is assumed that many Red Hat and CentOS systems currently deployed within the DoD are enforcing this policy (with the exception of national security systems and other high assurance systems). The default Targeted Policy restricts many services and processes (almost all network listening services) to a confined domain. Within the confined domain, these processes are only allowed to perform operations explicitly granted to them by the policy. This is usually limited to only the access that is required to perform the operations of the service. For example, the Apache web server is confined to only reading files that are associated with the web server. If the web server is compromised, the attacker is limited to reading and writing to only web-server files, even if the server is running as a service with root permissions. In this way the attack is limited in scope based on the confined policy.

Table 1 provides a short list of examples of common services that operate in the confined domain. RHEL 5 has 200 confined applications in the targeted policy, while RHEL 6 has increased to 330 confined processes.

Table 1. Examples of Confined services

|  |  |  |  |
| --- | --- | --- | --- |
| Apache webserver | Bind DNS | Mozilla | DHCPD |
| Squid proxy | Syslogd | CUPS Printing Svr | ntpd |
| MySQL database | PostgreSQL | portmap | snmpd |

To maintain user flexibility, all non-targeted policies are allowed to run in an unconfined domain, such as user created processes. These users and processes are still subject to standard DAC security mechanisms such as file access permissions. Unconfined domains are also prevented from allocating writable memory and executing it. This further reduces the applications vulnerability to buffer-overflow attacks.

Many National Security Systems rely on SELinux as the MAC enforcement mechanism to prevent unauthorized access to information and limit the scope of compromise that is possible with previously unknown (zero-day) vulnerabilities. Cross domain and guard systems are an example of systems requiring MAC assurance, one-way information flow, and covert back-channel analysis. These are all possible with SELinux enforcement and policy analysis.

# Analysis of McAfee HIPS Protection Capabilities

At the time of testing, McAfee HIPS contains a total of 83 rules for supported RHEL systems. For analysis purposes, the rules have been divided into 4 basic categories:

* **McAfee Self-Protection:** Agent shielding that protect against illegal access and disabling of McAfee protection services
* **Web-Server Protection:** Protection for Apache web server
* **Vulnerability Protection:** Protections against known vulnerabilities
* **Other:** Protections for unwanted and illegal system modifications and information disclosure.

Table 2. HIPS Rule Analysis

|  |  |  |
| --- | --- | --- |
| **RHEL 5/6 Rule Analysis** | | |
| **Total HIPS Rules Available:** | | **83** |
| McAfee Self Protection | | 3 |
| Web-Server Protection | | 19 |
| Known Vulnerability Protection | | 50 |
| Other (System Modification & Access) | | 11 |
| **Default Severity Setting** | | |
| **HIGH** | 11 | |
| **MEDUIM** | 50 | |
| **LOW** | 16 | |
| **DISABLED** | 6 | |

The full list of supported rule and how this report categorized them is contained in Appendix A of this document. It should be noted that during testing, none of the apache rules were observed as enforcing on the RHEL 6.1 (64-bit) system. It is currently unknown whether the cause for this is due to incompatibility with the released version or the 64-bit architecture.

# SELinux and McAfee HIPS security comparison

The Joint Interoperability Test Command (JITC), in a partnership with the DISA Field Security Operations, and the 177th Information Aggressor Squadron, conducted a test of McAfee HIPS on supported Non-Windows systems from 5-9 Dec 2011. During this testing, the team executed 5 test cases to compare the security provided by SELinux and McAfee HIPS, as well as additional testing to identify any negative impacts of attempting to enforce both policies at the same time. Test was conducted at the DoD Cyber Range, Stafford VA.

To accurately compare the 2 products, 3 systems were configured with an Apache web server and a MySQL database. All systems were installed with a known vulnerable web application known as DVWA (<http://code.google.com/p/dvwa/>). This configuration allowed for the simulation of a some common web application exploits. Web Application exploits were chosen as the comparison grounds due to both SELinux and HIPS having support for securing the Apache web server.

Table 3 shows the results of the testing. Note that the test cases are increasing in severity with the most sever being the ability to obtain a remote shell execution on the system.

Table 3. Web-Server exploit comparison

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **TTC#** | **Test Case** | **Alert and Protect** | | |
| **Default**  **SELinux** | **RHEL 5 HIPS** | **RHEL 6 HIPS** |
| 1 | File Inclusion to enumerate system users (view the /etc/passwd) | No | No | No |
| 2 | Cross Site Scripting (Script Tag) | No | Yes | No |
| 3 | Cross Site Scripting (Object, Applet, and Embed Tags) | No | No | No |
| 4 | Command Execution from browser (looking at /var/log/wtmp) | Yes | No | No |
| 5 | Using web vulnerability to obtain remote shell on server | Yes | No | No |

**Findings and Observations:**

1. In Test Case #1, neither McAfee HIPS or SELinux prevented a file inclusion vulnerability from reading the /etc/passwd files and enumerating the users on the system. This is due to apache requiring access to the /etc/passwd file to validate users on the system. All system users were successfully provided to the attacker.
2. In Test Case 2 and 3 a Cross-Site Scripting attack was launched through the web-server in order to serve malicious content to an unsuspecting client surfing the website. Although this results in a client-side compromise, it is due to vulnerabilities within the web application. In TC#2, McAfee HIPS successfully blocked the attack when using the <script> tag to compromise the client. The attack was then tested with the object, applet, and embed tags in TC#3 and was successfully able to compromise a client. SELinux did not alert or block on either of these test cases.
3. In Test Case 4 a vulnerability in the web application allowed for execution of a command on the web server through the attackers web browser. Numerous commands were executed ("ifconfig", "utmpdump /var/log/wtmp", "passwd", "nc"). SELinux successfully blocked and alerted on all instances. McAfee HIPS did not alert or block any of the commands.
4. In Test Case 5, the attackers were asked to obtain a remote shell on the system using any of the vulnerabilities present within the web-application. The attackers found 2 ways to obtain a shell:
   1. Using the command execution vulnerability to execute NetCat and connect remotely to the server.
   2. Redirecting */bin/bash* to */dev/tcp/host/port* to obtain a shell connection through a standard device

In both cases, SELinux prevented the attacker from gaining a remote shell and alerted on the offense. In both cases with SELinux disabled and McAfee HIPS enabled, the attack was successful and no alert was observed.

1. After testing was complete, the test team attempted to install McAfee HIPS on the system that had SELinux in "Enforcing" mode. The installation appeared to be successful, however the policy enforcement provided by HIPS was not functional. The system was reset. It was observed that the SELinux policy was still being enforced and the HIPS kernel modules were not enforcing the HIPS policy. No stability issues were witnessed. It should be noted that the McAfee products did not show any indication that it was not functioning properly due SELinux being enabled. No errors were immediately visible on the ePO server for the system under test.

**Recommendations:**

1. It is recommended that SELinux be enabled on systems that provide applications and services that are compatible with it. This should apply to the vast majority of systems, however if there are systems that require SELinux to be disabled, or that do not support it, they should be tested with McAfee HIPS and ultimately recommended to install it for added security and visibility.
2. Although no instability issues were observed, it is not recommended that HIPS be installed on any system that has another LSM module installed and enabled.

# Conclusion

As long as both McAfee HIPS and SELinux both utilize LSM as their policy enforcement mechanism, and neither one implements the ability to stack the other, they will be incompatible and not able to run simultaneously on the same system. While research has been done in the area of stacking modules, and technically the kernel is able to support the capability for security module vendors to implement stackable modules, it is not likely that we will see SELinux and McAfee HIPS able to enforce separate policies in the near future. With the current inability to enable both SELinux and McAfee HIPS on Red Hat Linux systems, it is necessary to choose which security framework provides the most benefits for the DoD in the form of both security and intrusion visibility. SELinux is the current Red Hat maintained and supported security framework and has been found to be far superior to McAfee HIPS in blocking and limiting attacks. Auditing of security alerts is typically conducted on the system itself, or via offloading to a syslog server. More research would need to be done to find out what current CND capabilities are maintained that could provide additional alerting and visibility to SELinux enabled systems. McAfee, while inferior in overall security, is able to utilize the current HBSS alert reporting capabilities to quickly provide alerts to the Computer Network Defense Service Providers and incident responders.

It is recommended that systems that currently utilize SELinux to enforce security policy, whether it be the default vendor targeted policy or a custom policy, should continue to enforce these policies and not install McAfee HIPS. It is recommended that McAfee HIPS only be installed on systems that do not have the ability to enable SELinux and maintain it in the enforcing mode.

# Works Cited

Mayer, F., MacMillan, K., & Caplan, D. (2007). *SELinux by Example.* Upper Saddle River, NJ: Prentice Hall.

Quaritsch, M., & Winkler, T. (2004). *Linux Security Modules Enhancements: Module Stacking Framework and TCP State Transition Hooks for State-Driven NIDS.* Graz, University of Technology, Austria.

Red Hat. (2010). *Security Enhanced Linux.* Retrieved December 02, 2011, from Red Hat Documentation: http://docs.redhat.com/docs/en-US/Red\_Hat\_Enterprise\_Linux/6/html/Security-Enhanced\_Linux/chap-Security-Enhanced\_Linux-Targeted\_Policy.html

Various. (n.d.). *Damn Vunerable Web App*. Retrieved December 5, 2011, from Google Code: http://code.google.com/p/dvwa/

Web Application Consortium. (2011, December 10). *WASC Threat Classification.* Retrieved from The Web Application Security Consortium: http://projects.webappsec.org/w/page/13246978/Threat%20Classification

**Appendix A. HIPS Rules as of this writing**

|  |  |
| --- | --- |
| **McAfee Self Protection Rules - 3** | |
| Linux Agent Shielding - File Mod | Linux Agent Shielding - Package Mod |
| Linux Agent Shielding - Module Access |  |
| **Web Server Protections - 19** | |
| Linux Apache Shielding - Server File Access | Linux Apache Envelope - File Access |
| Linux Apache Shielding - Server File Mod | Linux Apache Envelope - File Modification |
| Linux Apache Shielding - Docs File Access | Linux Apache Envelope - File Access by CGI |
| Linux Apache Shielding - Docs File Mod | Linux Apache Envelope - File Mod by CGI |
| Linux Apache Shielding - CGI File Access | Apache SSI File Extension Request |
| Linux Apache Shielding - CGI File Mod | Apache Artificially Long Slash Path |
| Linux Apache Shielding - Log File Access | Apache test-cgi Directory Listing |
| Linux Apache Shielding - Log File Mod | Apache Cross-Site Scripting |
| Linux Apache Shielding - Configuration File Access | Apache mod\_perl 'Apache::Status' and 'Apache2::Status' Cross-Site Scripting Vulnerability |
| Linux Apache Shielding - Conf File Mod |  |
| **Known Vulnerability Protection - 50** | |
| Linux - Vulnerability in kdebase | Linux CUPS Insecure Temporary File Creation Vulnerability |
| Linux - KDE kcheckpass Insecure Lock File Creation Vulnerability | Linux - Vulnerability in Coolkey Could Allow Local Users to Overwrite Arbitrary Files |
| Linux - Texinfo Temporary File Creation Vulnerability | Xfig in Debian GNU/Linux Could Allows Read and Write Arbitrary Files |
| Linux - Vulnerability in PHP that could allow bypass of security restrictions | Linux - Vulnerability in Fence Could Allow Local Users to Overwrite Arbitrary Files |
| Linux - Vulnerability in util-linux could allow Elevation of Privileges | Linux - Linux Kernel 2.6 UDEV Vulnerability Could Allow Elevation of Privileges |
| Linux - Vulnerability in net-snmp Temporary File Creation | Linux - MySQL Aborted Bug Report Insecure Temporary File Creation Vulnerability |
| Linux - Vulnerability in cvs could allow Elevation of Privileges | Linux - chm2pdf Insecure Temporary File Creation Vulnerability |
| Linux - Vulnerability in sysreport could allow Elevation of Privileges | Linux - gpicview Insecure Temporary File Creation Vulnerability |
| Linux - Vulnerability in gdb could allow Elevation of Privileges | Linux - sysstat Insecure Temporary File Creation Vulnerability |
| Linux - Vulnerability in mysql could allow Elevation of Privileges | Linux - am-utils Insecure Temporary File Creation Vulnerability |
| Linux - Vulnerability in kdelibs could allow Denial of Service | Linux - cman/fence Insecure Temporary File Creation Vulnerability |
| Linux - Vulnerability in lm\_sensors could allow Elevation of Privileges | Linux - moodle Insecure Temporary File Creation Vulnerability |
| Linux - Vulnerability in sharutils could allow Elevation of Privileges | Linux - acroread Insecure Temporary File Creation Vulnerability |
| Linux - Vulnerability in mysql could allow Elevation of Privileges via symlink attacks | Mozilla Firefox Temporary File Vulnerability |
| Linux - Vulnerability in vim could allow Elevation of Privileges | sblim-sfcb 'genSslCert.sh' Insecure Temporary File Creation Vulnerability |
| Linux - Vulnerability in cron could allow Denial of Service (DoS) | Enomaly ECP/Enomalism Insecure Temporary File Creation Vulnerability |
| Linux - Vulnerability in Mutt mail client could allow local users to overwrite arbitrary files | Skype for Linux Insecure Temporary File Creation Vulnerability |
| Linux - VMware Shielding - Configuration File Modification | xscreensaver temporary file Vulnerability |
| Linux - init.d XFS Script chown Race Condition Vulnerability allows Elevation of Privileges | Wyrd uses insecure temporary file Vulnerability |
| Linux - Qemu-dm.debug in Xen symlink attack vulnerability allows local users to overwrite arbitrary files | openswan Insecure auxiliary /tmp file usage Vulnerability |
| Linux - Gnu Screen Temporary File Creation Vulnerability | sealert in setroubleshoot 2.0.5 insecure temporary file handling vulnerability |
| Linux - Xen Temporary truncate arbitrary files Vulnerability | GNU Emacs vcdiff Insecure Temporary Files Vulnerability |
| Linux - Kernel PRCTL core dump handling privilege escalation Vulnerability | Puppet Insecure Temporary Files Vulnerability |
| Linux - Kernel exit\_notify() CAP\_KILL Verification Vulnerability | Ntop Red Hat Initialization Script Insecure Temporary File Creation |
| Apache WordPress 'wp-login.php' Admin Password Reset Security Bypass Vulnerability | Merkaartor Insecure Temporary File Creation Vulnerability |
| **Other (System Modification & Access) - 11** | |
| Linux - Link to System File Created | Linux - SUID/SGID File Creation |
| Linux - ypxfrd directory traversal | Linux - SGID File Modification |
| Linux - PCMCIAD File Corruption | Linux - Startup File Mod |
| Linux - at Command file Deletion | Linux - Startup File Creation or Deletion |
| Linux - Password File Mod | Linux - Startup File Attr Mod |
| Linux - VMware Shielding - Device File Access |  |