



The Path to Multi-Level Security in Red Hat Enterprise Linux

By Chris Runge

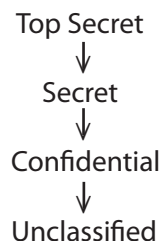
Organizations requiring Multi-Level Security (MLS) face many challenges. Implementing MLS is an expensive process, and mainstream operating systems do not include traditional MLS and Mandatory Access Control (MAC) capabilities. Most of the trusted operating systems that did provide these capabilities have failed and exited the market, leaving the need for a new solution.

This paper is an introduction to Multi-Level Security, and examines the problems associated with past implementations. It provides a road map for MLS in Red Hat® Enterprise Linux®, and demonstrates how Security-Enhanced Linux, in conjunction with Red Hat Enterprise Linux, provides this new, more capable and more flexible solution.

A brief introduction to MLS

MLS is an implementation of MAC that focuses on confidentiality. A Multi-Level Security operating system is able to enforce the separation of multiple classifications of information as well as manage multiple users with varying levels of information clearance.

The military and intelligence community must handle and process various hierarchical levels of classified information. At the high-end there is Top Secret, followed in turn by Secret, Confidential, and Unclassified:



These sensitivity levels base themselves on the damage to national security that could occur if certain information leaked out; the higher the sensitivity level, the greater the potential damage.

Access to classified information is based on a clearance, a level of trust placed in an individual after a proper background check. These clearances are associated with the hierarchical sensitivity levels mentioned previously. For example, a person with a Secret clearance can access Secret, Confidential, and (obviously) Unclassified data, but cannot access information that has been classified Top Secret.

Further limiting access is the principle of “need to know.” Just because individuals have a Top Secret clearance does not necessarily mean they have access to all Top Secret data. They may be cleared only to access information they have a “need to know” to do their job. For example, a person working on satellites may not have a need to know information about weapons systems. “Need to know” is enforced through categories or compartments, which are -- unlike sensitivity levels -- non-hierarchical in nature. Example categories could be “US Only” or code words like “Alpha” or “Bravo.” To gain access to information classified as Top Secret/Alpha/Bravo, a person would need to have a Top Secret clearance as well as have been granted access to both the Alpha and Bravo compartments.

A multi-user system within the military and intelligence community may have users with different levels of clearance, each trying to access information with different levels of classification. The operating system used in these environments plays a critical role. It must allow users to access the data for which they have been cleared while ensuring they do not access information they are not authorized to review. Because of the nature of the information present-- and the potentially grave consequences that could occur should that information be leaked-- the operating system must enforce these restrictions irrespective of the actions of programs, users, or administrators on that system. A Multi-Level Security operating system is one that permits the appropriate information flow and enforces these restrictions.

The classic model for MLS is Bell-LaPadula (BLP). The fundamental precepts of this model can be summarized in the phrase “no read up, no write down.” As discussed previously, a user on the system must not be allowed to “read up”--that is, read information that is of a higher level than the user’s clearance. For example, a user with Confidential clearance must not read information classified as Secret or Top Secret. In addition, users must not be able to read information in compartments to which they do not have access. These restrictions also extend to programs run by that user on the system.

Besides “no read up,” users and programs on the system must not be able to write information classified at one level down to a lower level. For example, a user with Top Secret clearance, who by nature of that clearance has access to Top Secret, Secret, Confidential, and Unclassified information, must not be able to write information that has been classified as Top Secret into a document that is classified at a Secret or some other lower level. To allow such an action would compromise that data, as users with a Secret clearance would now have access to Top Secret information.

MLS/BLP is one implementation of MAC. On a MAC operating system a security policy on the system is the final arbiter for all access control decisions. A flawed, compromised, or misconfigured program must not be able to work around the security policy, nor the accidental or intentional actions of a user or even a system administrator. A Discretionary Access Control

(DAC) model functions differently as users and programs have discretion over their objects on the system, such as their files. For example, on a DAC system users own files that they create (such as those in their home directory) and may choose to allow others to read and/or write to those files. Programs run as that user have the same level of privilege and therefore may also read and write to those files. On a MAC system the security policy on the system, not the user, makes the final determination as to who and what can read, write, and execute certain files. MAC eliminates the need for an all powerful superuser who has access to everything in a DAC model. Attempts to enforce the restrictions of MLS/BLP without MAC would fail, as system administrators would be able to override those restrictions.

Past approaches to MLS

Traditionally, MLS through the BLP model has been implemented in what are commonly known as trusted operating systems. The operating system is trusted to enforce the strictures of information flow amongst multiple users and multiple security levels on the system. Examples of trusted operating systems include Trusted Tru64, Trusted HP-UX, Trusted AIX, and Trusted Solaris.

The commercial availability of these trusted operating systems has been less than successful, both for the manufacturers of these products and for those who attempted to implement them. The one major exception to this rule has been the comparative success of Trusted Solaris within the military and intelligence community. There are several reasons for their lack of adoption, key among them the fact that these trusted operating systems have been separate from their mainstream commercial counterparts, lagging behind their cousins in core operating system features and capabilities. For example, the latest version of Trusted Solaris, version 8, lacks capabilities like containers, DTrace, and ZFS that are present in Solaris 10. Also, the ecosystem of certified hardware and applications is smaller, as is the pool of available expertise to implement and administer these solutions properly. All of these reasons, in addition to the initial cost of the software itself, make these systems expensive to deploy and maintain.

The viability of the trusted operating system approach has been complicated by the fact that the BLP model does not lend itself well to many practical implementations, even within the military and intelligence community. For example, there may be instances where certain aspects of classified information may need to be released, or written down, to a lower security level. One example is highly classified intelligence at the Top Secret level that is used to provide targeting instructions to the war fighter, who may only possess a Secret clearance. Real-world events force the adoption of workarounds in system design and deployment that subvert the BLP model.

In the private sector there are end users who would benefit from the confidentiality of information that MLS guarantees, for example, to ensure compliance with regulations such as Sarbanes-Oxley and HIPAA. However, BLP as a model of information flow does not map itself well to commercial business practices. Previous efforts to implement trusted operating systems outside of the military and intelligence community, nearly universally, failed.

Despite these challenges MLS is still a requirement for many deployments. Faced with aging systems, the decreasing availability of commercial solutions, and doubt about the viability of those that remain, these organizations are ready for a new approach.

A new approach: Red Hat Enterprise Linux 4

The release of Red Hat Enterprise Linux 4 in February 2005 marked the first time MAC was delivered as a core feature of a commercially available, mainstream operating system. This was provided through the inclusion and full support of Security-Enhanced Linux, initially started as a research project by the NSA to add MAC to Linux. Security-Enhanced Linux, or SELinux, is developed within the open source community and has been incorporated into the upstream 2.6 Linux kernel.

SELinux in Red Hat Enterprise Linux 4 provides Type Enforcement, another approach to MAC besides the MLS/BLP model. Whereas the BLP model of Multi-Level Security is focused on data confidentiality at the sake of integrity, Type Enforcement does not force users to make that compromise. SELinux and Type Enforcement allow users to build systems that provide high levels of both data confidentiality and integrity. Type Enforcement's approach is more flexible, implementing different policies to achieve and enforce security and functionality requirements.

The key principle of BLP is "no read up, no write down;" Type Enforcement adopts the principle of least privilege. According to this principle, an application is given just enough permission to function as intended, but no more. For example, a particular deployment of a web server may require that it be able to listen on port 80 on the external network interface, allow users to view web pages within a particular directory (e.g. /var/www/html), and run certain CGI scripts. The security policy on the system would allow these activities as well as allow the necessary mechanics required for the web server to operate, such as reading (not writing) its configuration file, and reading and writing its log files. Denied by the security policy on the system would be other activities such as listening on different ports or different network interfaces, displaying data in other directories, and accessing firewall rules.

Under a Type Enforcement model, applications run in separate areas, known as domains, isolated from one another and from the underlying operating system. These domains are defined by the security policy on the system. A flaw or misconfiguration in an application protected by Type Enforcement is isolated within that application's domain.

To understand the benefits of Type Enforcement, consider a web server on a traditional Discretionary Access Control system. Because the web server needs to listen on a privileged port it must run as the superuser. A compromise of the web server grants an attacker these same superuser privileges, possibly affecting the integrity of the entire system, including all of the data and applications that reside on that system. However, given a system with Type Enforcement, such as on Red Hat Enterprise Linux 4 with SELinux enabled, a compromise of the web server would limit the attacker to those actions permitted by the security policy, namely reading certain web pages, running certain CGI scripts, and reading (but not writing) the web server's configuration file. The attacker could not use the compromised application as a springboard to escalate their privileges to other areas of the system.

Red Hat recognized the advantages of MAC have a wider application than the traditional customer base of the trusted operating systems (consider the benefits of the web server example to an ISP or e-commerce site). Avoiding the mistakes of the past, Red Hat intentionally made SELinux a core feature of Red Hat Enterprise Linux 4. There is no separate Red Hat Enterprise Trusted Linux product, nor is the purchase of "Trusted Extensions" or a similar add-on product required in order to implement and use SELinux. Customers have access to a comprehensive ecosystem of certified hardware and third-party applications, as well as the latest advantages in core operating system functionality. These are the benefits of deploying a rapidly growing, mainstream operating system.

By default, Red Hat Enterprise Linux has SELinux enabled and implements a targeted policy designed to draw a balance between usability on the one hand and increased security on the other. The targeted policy confines several network-listening daemons--services such as web servers, DNS servers, and mail servers that are constantly under attack--while generally not impacting an end user's ability to run other applications and otherwise use the system.

Because SELinux and Type Enforcement offer a flexible security model, allowing one to tailor the policy to the actual functional needs and security requirements of users on the system, it allows for the implementation of other policies. Users may choose to disable SELinux completely and run in a traditional Discretionary Access Control model, or they can have the ability to use an optional strict policy, a superset of the targeted policy that confines many more applications and places additional limits on what end users can do on the system.

The path forward: Red Hat Enterprise Linux 5

Where does this leave MLS and the Bell-LaPadula model? While Type Enforcement, owing to its flexibility, is arguably better suited for most uses, there remain several critical instances where traditional MLS is still required. These typically involve a large number of classification levels with multiple permutations of several different compartments and sensitivity levels and multiple users with varying levels of clearance (for instance, a Top Secret/Alpha, a Top Secret/Bravo, a Top Secret/Alpha/Bravo, and a Top Secret/Alpha/Bravo/US Only).

In order to enable these environments, Red Hat Enterprise Linux 5 will add support for MLS using the Bell-LaPadula model in conjunction with Type Enforcement. This MLS policy will be one of the available policy options, in addition to the default targeted policy and optional strict policy currently available for Red Hat Enterprise Linux 4. The integration of MLS with Type Enforcement provides the integrity controls that were previously lacking in trusted operating systems that only utilized the Bell-LaPadula model. This will give users the benefits of both approaches to MAC in a single, unified, and analyzable policy.

SELinux will serve as the foundation for the MLS policy. As with the original work that was done to implement SELinux in Red Hat Enterprise Linux 4, the initial developments are being done within the Fedora Project. Fedora Core 5, currently in testing and scheduled for release in March 2006¹, will have much of the core infrastructure in place necessary to support the MLS policy¹. This will enable users and developers to begin testing and exercising the code paths used by MLS. The approach is to refine these capabilities so that they are production-ready for Red Hat Enterprise Linux 5.

¹ For the latest Fedora Core 5 schedule, refer to <http://fedora.redhat.com/About/schedule/>

It is important to note that, just as with the advent of MAC and Type Enforcement in Red Hat Enterprise Linux 4, the basis for the new MLS support will be in Red Hat Enterprise Linux 5; as before, there will be no separate “Red Hat Enterprise Trusted Linux.” The presence of these capabilities in a mainstream operating system will help to avoid the mistakes of the past that were characteristic of the trusted operating system approach. Customers will no longer have to choose between having a fully-featured modern operating system with an ecosystem of certified applications and hardware platforms or a trusted operating system that has MLS and other MAC capabilities. In fact, because customers can determine the security policy they wish to implement for a given system—Discretionary Access Control, MAC with Type Enforcement and the targeted policy, MAC with Type Enforcement and a strict policy, or MAC with MLS using the Bell-LaPadula model—the same operating system can be deployed across a range of varying operational and security requirements.

Red Hat Enterprise Linux has achieved Common Criteria evaluation on numerous architectures. Currently, Red Hat Enterprise Linux 4 is the latest version to be evaluated for EAL4+ with the Controlled Access Protection Profile (CAPP). Red Hat Enterprise Linux 5, scheduled for release in late 2006, is officially “in evaluation” for EAL 4+ with CAPP, Labeled Security Protection Profile (LSPP), and Role-Based Access Control Protection Profile (RBACPP)². Common Criteria EAL 4/LSPP replaces the older TCSEC Orange Book B1 standard for MLS operating systems. The addition of MLS capabilities in conjunction with the Common Criteria evaluation enables Red Hat Enterprise Linux 5 to be the foundation for a solution that meets the requirements of DCID (Director of Central Intelligence Directive) 6/3 Protection Level 4 (PL4) for Top Secret and Below Interoperability (TSABI). This represents the highest level of security capability to date for a Linux distribution, formerly the province of a select few of the trusted operating systems.

Getting started

System integrators and developers need not wait until the commercial availability of Red Hat Enterprise Linux 5, scheduled for release in late 2006, to begin the migration of their MLS applications. Since SELinux will be used as the foundation for the MLS capabilities, an understanding of how SELinux works and how to write SELinux policy is a necessary first step. Red Hat Enterprise Linux 4 and Fedora Core 3 and above include SELinux as a core feature of the operating system and can be used to gain hands-on experience with the technology. Formal training is also available from Red Hat, including an “Introduction to SELinux and Red Hat Targeted Policy”³ and a more advanced class on “Red Hat Enterprise SELinux Policy Administration.”⁴

In addition to the particulars of SELinux policy writing, Red Hat offers training and support for application developers currently working in a proprietary UNIX environment desiring assistance with porting and developing applications for Red Hat Enterprise Linux. This includes a class on “Red Hat Linux Application Development and Porting”⁵ and Premium Developer Support⁶. Professional Services to assist with the migration and porting effort are also available from Red Hat and its partners.

As mentioned earlier, Fedora Core 5 will form the basis for Red Hat Enterprise Linux 5. Currently available as a test release, Fedora Core 5 contains the infrastructure necessary for supporting MLS environments. For more information and to download this release, along with the final Fedora Core 5 release when available, visit <http://fedoraproject.org>. Moving at a more rapid pace, the latest developments in the targeted, strict, and MLS policies for SELinux are available at <ftp://people.redhat.com/dwalsh/SELinux/Fedora>.

² http://niap.nist.gov/cc-scheme/in_evaluation.html#r

For the Red Hat, IBM, and Trusted Computer Solutions press release see http://www.redhat.com/en_us/USA/home/company/news/prarchive/2005/most_secure_linux.html

For the latest information on Red Hat Enterprise Linux and Common Criteria visit http://www.redhat.com/en_us/USA/home/solutions/government/commoncriteria/

³ For more information see <https://www.redhat.com/training/security/courses/rhs427.html>

⁴ For more information see <https://www.redhat.com/training/security/courses/rhs429.html>

⁵ For more information see <https://www.redhat.com/training/developer/courses/rhd256.html>

⁶ For more information see <https://www.redhat.com/support/offerings/premium.html>



Red Hat has also started a mailing list for those who wish to participate more fully in the ongoing engineering efforts to incorporate MLS capabilities into Red Hat Enterprise Linux 5. This list is open to all interested parties, who may subscribe at <https://www.redhat.com/mailman/listinfo/redhat-lspp>.

Finally, it should be noted that many scenarios that used traditional MLS in the past might in fact be better served by the Type Enforcement capabilities already provided in Red Hat Enterprise Linux 4. Type Enforcement provides a more flexible approach than the Bell-LaPadula model of MLS, as the policy can be written to reflect actual functional and security requirements.

Conclusion

Organizations requiring MLS have faced many challenges. Because the capabilities necessary for enabling MLS have not been previously offered in a mainstream operating system, these organizations have been forced to use expensive trusted operating systems, which lacked many of the basic capabilities, hardware support, and third-party software support of their mainstream cousins. The situation has worsened, as most of these trusted operating systems have failed and exited the market, leaving organizations unsure of where to turn next as existing deployments are reaching their end-of-life and as new requirements emerge.

With the incorporation of SELinux in Red Hat Enterprise Linux 4, Red Hat took an unprecedented step by adding MAC into a growing, mainstream operating system. Its flexible Type Enforcement and targeted security policy capabilities extended the benefits of MAC to a wider class of users.

Red Hat Enterprise Linux 5 will use the SELinux foundation introduced in Red Hat Enterprise Linux 4 to enable MLS for the critical scenarios that require this security model. This will enable users of the trusted operating systems to take advantage of the benefits that Red Hat Enterprise Linux has offered UNIX users in the past, namely an operating system that has UNIX-like capabilities but with the ability to run on lower-cost, higher-performance, industry-standard hardware. Ultimately these organizations will gain far greater flexibility than in the past by being able to run on modern hardware platforms based on a number of architectures from a number of mainstream providers. In addition they will be able to use the same operating system for a range of deployments, from those that require MLS to those where the targeted policy or even Discretionary Access Control is sufficient.

Additional resources

Dr. Rick Smith. "Introduction to Multilevel Security."
<http://www.cs.stthomas.edu/faculty/resmith/r/mls/index.html>

James Morris. "An Overview of Multilevel Security and LSPP under Linux."
http://www.livejournal.com/users/james_morris/5020.html