www.ijecs.in
International Journal Of Engineering And Computer Science ISSN:2319-7242
Volume - 3 Issue -9 September, 2014 Page No. 8265-8269

Survey on Linux Security and Vulnerabilities

Ashvini T. Dheshmukh[1], Dr. Parikshit. N . Mahalle [2]

[1] *Department of Computer Engineering , Smt. Kashibai Navale college of Engineering, University of Pune, Maharashtra, India*

[2] *Department of Computer Engineering , Smt. Kashibai Navale college of Engineering, University of Pune, Maharashtra, India*

Abstract*: This paper is intended as overview of Linux security and different threads to server, network and workstation. SELinux, which is an implementation of Linux Security Modules (LSM), implements several measures to prevent unauthorized system usage. Security is a very broad concept, and so is the security of a system. All too often, people believe that a system is way more secure that it in practice is, but the biggest problems is still the human factor of the users; the possibility of careless or malicious users are commonly overlooked. Finally this paper concludes with providing list of some various common vulnerability, attacks and countermeasures.*

Keywords: SELinux, MAC, sshd , cron, Telnet.

1. Introduction

Because of the increased reliance on powerful, networked computers to help run businesses and keep track of our personal information, entire industries have been formed around the practice of network and computer security. Enterprises have solicited the knowledge and skills of security experts to properly audit systems an d tailor solutions to fit the operating requirements of their organization. Because most organizations are increasingly dynamic in nature, their workers are accessing critical company IT resources locally and remotely, hence the need for secure computing environments has become more pronounced.

The standard "Unix way" of providing authentication and authorization is not very well suited for more complicated and dynamic environments. The available Discretionary Access Control mechanisms gives the same rights to all users in a certain group, and all processes created by a user have exactly the same privileges. The acquired permissions can also be transferred to other subjects, and so a flaw in one software can lead to all the users' data being compromised. Some system wide permission restrictions can be enforced by using several user groups, limiting the use of certain soft-ware to users belonging to that group, but each user process still ahve all the permissions of all the groups that the owning user belongs to[1]. Government agencies, among other similar organizations, need a more advanced way of defining system security policies. That is why the National Security Agency (NSA) began developing, for internal use, their own set of patches to the Linux kernel. These patches became known as Security-Enhanced Linux (SELinux) and was later released under the GPL and included in the main Linux kernel tree.

Nowadays SELinux is a security module for the Linux Security Modules framework. This paper is an overview of the SELinux features and how it changes the security con-figuration aspect of Linux. The architecture that SELinux implements, the Flask architecture, and its components are described in next section. A brief description of the SELinux LSM module is given in next section. This paper is intended as overview of Linux security and different threads to server, network and workstation. SELinux, which is an implementation of Linux Security Modules (LSM), implements several measures to prevent unauthorized system usage. Security is a very broad concept, and so is the security of a system. All too often, people believe that a system is way more secure that it in practice is, but the biggest problems is still the human factor of the users; the possibility of careless or malicious users are commonly overlooked. Finally this paper concludes with providing list of some various common vulnerability, attacks and countermeasuresThere is plenty of in-depth documentation of the internals in the sources of this paper [2][3] .

2. Motivation

Because of the increased reliance on powerful, networked computers to help run businesses and keep track of our personal information, entire industries have been formed around the practice of network and computer security. Enterprises have solicited the knowledge and skills of security experts to properly audit systems and tailor solutions to fit the operating requirements of their organization. Because most organizations are increasingly dynamic in nature, their workers are accessing critical company IT resources locally and remotely, hence the need for secure computing environments. Unfortunately, many organizations (as well as individual users) regard security as more of an afterthought, a process that is overlooked in favor of increased power, productivity, convenience, ease of use, and budgetary concerns. Proper security implementation is often enacted postmortem — after an unauthorized intrusion has already occurred. Taking the correct measures prior to connecting a site to an entrusted network, such as the Internet, is an effective means of thwarting many attempts at intrusion.

Although not the dominant operating system on the Internet, Linux is quite prevalent, considering that the overwhelming majority of servers running web services, email services, and name services all depend on other open-source code that works with Linux. And this is where the trouble begins. So we need to secure Linux operating system.

3.Related Work

3.1 SELinux

SELinux started as a security research project at NSA, together with Secure Computing Corporation and the University of Utah, to demonstrate the benefits of mandatory access control over the user/group schema. Today SELinux is included in the mainstream Linux kernel as a security module in the LSM framework. SELinux implements a flexible mandatory access control (MAC) architecture in the major subsystems of the kernel and provides a mechanism to en-force the separation of information based on confidentiality and integrity requirements [2].

3.1.1 Basic Architecture

NSA tried to get their SELinux patches. development branch kernel back in 2001, but Linus Torvalds rejected the proposal since there were other similar ongoing projects at the same time. A more general solution was needed so that the kernel would be able to support as many security architectures and implementations as possible, with-out sticking too much to the ideas of any specific implementation

3.1.2 Linux security modules

To support various security models, an interface "Linux Security Module Interface" was proposed [4] by Crispin Cowan. The Linux Security Modules framework development got contributions from huge corporations, such as IBM and SGI, and naturally NSA. In 2006, the only widely used LSM module included in the mainstream kernel was SELinux, but Torvalds still wanted to keep the door open for other implementations1 and so SELinux was finally included in the mainstream 2.6 kernel as a security module in late 2003.

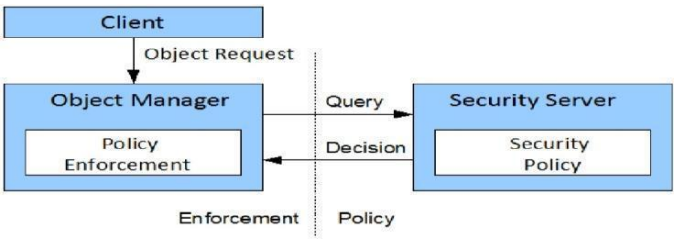

Fig.1. Flask Architecture

3.1.3 Flask architecture and concepts

The very flexible MAC architecture used in SELinux is Flask [4], which was derived from a micro-kernel based operating system named Fluke. Flask clearly separates the security policy from the enforcement mechanism (Fig. 1).

All subjects (processes) and objects (files, sockets, etc.) have a set of security attributes, referred to as the security context of the object. The attributes depend on the specific security policy in use, but generally contain the user id, a role and type enforcement domain.

Instead of working with the security context all the time, the security server maintains a mapping between security at-tribute sets and security identifiers (SIDs). When the object manager, the enforcing component, request labeling or ac-cess decisions, it typically passes a pair of SIDs to the secu-rity server, which looks up the related security contexts and makes the decision based on the policy in use.

Polyinstantiation is used when a certain resource needs to be shared by many clients. Such a resource could be the /tmp directory or the TCP port space. Filenames or port numbers might disclose some information about the owning process, and shared directories are subject to race-condition attacks. With polyinstantiation, each user can only see his or her own version of the resource based on username and/or security context.

The security server exists to provide policy decisions, map security contexts to SIDs, provide new SIDs and manage the Access Vector Caches (AVC), It usually also provides means for loading policies and it keeps track of which subjects can access its services.

3.1.4 Type Enforcement

The SELinux Type Enforcement (TE) model differs slightly from traditional models; by using the security class information provided by the Flask architecture and using a single type attribute for both processes and objects. This effectively means that a single matrix is used to specify the access interaction between different types, and objects of the same type can be treated differently if their associated security classes differ. Users are not directly bound to security types, but instead RBAC is used. Process transition rules are based on the current process domain, while types created through object transition rules are based on the creating process domain2 (the security type of the process identifier), the object security class and the type of the related object (e.g. parent directory for files). A process cannot change its domain during execution. Transition rules and access vectors have default policies for cases where a rule is not explicitly defined; Allowed transitions are not audited unless defined by audit allow rule, transitions are allowed only if explicitly allow rule and denied transitions are audited. All kinds of access vectors can have rules, including allow, audit allow, don't audit etc.

3.1.5 Role-Based Access Control

Role-based access control (RBAC) is used to define a set of roles that can be assigned to users. It is a more flexible model than standard DAC or MAC, and can simulate both of them using suitable rules. SELinux further extends the RBAC model to restrict roles to specified TE domains, and roles can be arranged in a priority hierarchy. Restricting roles to certain security domains allows most of the security decisions to be made through the TE configuration. The security context of a process contains a role attribute and also, while they are not actually applied, to objects. Role transitions are usually limited to a few TE domains to limit transitions to defined programs and users that need the ability, thus reducing the impact of malicious code being executed.

3.1.6 Multilevel security (MLS)

While type enforcement is the most important provider of mandatory access control, there might sometimes be a need for traditional MLS. SELinux option-ally provides MLS abilities, which allows defining a hierarchical "sensitivity" level and categories to objects and subjects (processes). Subjects and objects can have a range of security levels (e.g. directories might contain files with different security levels and some "trusted processes" might need to downgrade information)

defined when needed, but usually only one level is used. Any MLS defined constraints are enforced in addition to the TE policy, which means that checks must pass both of them for access to be granted.

3.1.7 User Identity

The "Unix" way of representing user identities using UIDs and GIDs is insufficient for SELinux, since changing a user role (e.g. su) involves changing the UID, which means that the actions following are actually performed as the other user and not just as the same user in another role. This makes auditing and accounting very difficult. SELinux user identity attribute is persistent in the security context, which is independent of the current UID. This means that SELinux policies can be enforced without affecting compatibility with Linux DAC permissions. Only a limited number of programs, like login, sshd and cron, need the ability to change the User Identity, so it is usually restricted to their respective TE domains. Depending on security configuration, the programs may or may not be able to change the user identity more than once. Allowing programs started from cron to change their user identity, for example, impacts accountability.

4. Threats To Network Security

Bad practices when configuring the following aspects of a network can increase the risk of attack.

4.1 Insecure Architectures

A misconfigured network is a primary entry point for unauthorized users. Leaving a trust-based, open local network vulnerable to the highly-insecure Internet is much like leaving a door ajar in a crime-ridden neighborhood nothing may happen for an arbitrary amount of time, but eventually someone exploits the opportunity.

4.2 Broadcast Networks

System administrators often fail to realize the importance of networking hardware in their security schemes. Simple hardware such as hubs and routers rely on the broadcast or non-switched principle; that is, whenever a node transmits data across the network to a recipient node, the hub or router sends a broadcast of the data packets until the recipient node receives and processes the data. This method is the most vulnerable to address resolution protocol (ARP) or media access control (MAC) address spoofing by both outside intruders and unauthorized users on local hosts.

4.3 Centralized Servers

Another potential networking pitfall is the use of centralized computing .A common cost-cutting measure for many businesses is to consolidate all services to a single powerful machine. This can be convenient as it is easier to manage and costs considerably less than multiple-server configurations. However, a centralized server introduces a single point of failure on the network. If the central server is compromised, it may render the network completely useless or worse, prone to data manipulation or theft. In these situations,a central server becomes an open door which allows access to the entire network.

5. Threats To Server Security

Server security is as important as network security because servers often hold a great deal of an organization's vital information. If a server is compromised, all of its contents may become available for the cracker to steal or manipulate at will. The following sections detail some of the main issues.

5.1 Unused Services and Open Ports

A full installation of Red Hat Enterprise Linux 6 contains 1000+ application and library packages. However, most server administrators do not opt to install every single package in the distribution, preferring instead to install a base installation of packages, including several server applications. A common occurrence among system administrators is to install the operating system without paying attention to what programs are actually being installed. This can be problematic because unneeded services may be installed, configured with the default settings, and possibly turned on. This can cause unwanted services, such as Telnet, DHCP, or DNS, to run on a server or workstation without the administrator realizing it, which in turn can cause unwanted traffic to the server, or even, a potential pathway into the system for crackers.

5.2 Unpatched Services

Most server applications that are included in a default installation are solid, thoroughly tested pieces of software. Having been in use in production environments for many years, their code has been thoroughly refined and many of the bugs have been found and fixed. However, there is no such thing as perfect software and there is always room for further refinement. Moreover, newer software is often not as rigorously tested as one might expect, because of its recent arrival to production environments or because it may not be as popular as other server software.

Although these mechanisms are an effective way of alerting the community to security vulnerabilities, it is up to system administrators to patch their systems promptly. This is particularly true because crackers have access to these same vulnerability tracking services and will use the information to crack unpatched systems whenever they can. Good system administration requires vigilance, constant bug tracking, and proper system maintenance to ensure a more secure computing environment.

5.3 Inattentive Administration

Administrators who fail to patch their systems are one of the greatest threats to server security. According to the SysAdmin, Audit, Network, Security Institute (SANS), the primary cause of computer security vulnerability is to "assign untrained people to maintain security and provide neither the training nor the time to make it possible to do the job." This applies as much to inexperienced administrators as it does to overconfident or a motivated administrators. Some administrators fail to patch their servers and workstations, while others fail to watch log messages from the system kernel or network traffic. Another common error is when default passwords or keys to services are left unchanged.

For example, some databases have default administration passwords because the database developers assume that the system administrator changes these passwords immediately after installation. If a database administrator fails to change this password, even an inexperienced cracker can use a widely-known default password to gain administrative privileges to

the database. T hese are only a few examples of how inattentive administration can lead to compromised servers

5.4 Inherently Insecure Services

Even the most vigilant organization can fall victim to vulnerabilities if the network services they choose are inherently insecure. For instance, there are many services developed under the assumption that they are used over trusted networks; however, this assumption fails as soon as the service becomes available over the Internet — which is itself inherently untrusted. One category of insecure network services are those that require unencrypted usernames and passwords for authentication. Telnet and FTP are two such services.

If packet sniffing software is monitoring traffic between the remote user and such a service usernames and passwords can be easily intercepted. Inherently, such services can also more easily fall prey to what the security industry terms the man-inthe- middle attack. In this type of attack, a cracker redirects network traffic by tricking a cracked name server on the network to point to his machine instead of the intended server. Once someone opens a remote session to the server, the attacker's machine acts as an invisible conduit, sitting quietly between the remote service and the unsuspecting user capturing information. In this way a cracker can gather administrative passwords and raw data without the server or the user realizing it.Another category of insecure services include network file systems and information. Services such as NFS or NIS, which are developed explicitly for LAN usage but Are, unfortunately, extended to include WANs (for remote users). NFS does not, by default, have any authentication or security mechanisms configured to prevent a cracker from mounting the NFS share and accessing anything contained therein. NIS, as well, has vital information that must be known by every computer on a network, including passwords and file permissions, within a plain text ASCII or DBM (ASCII-derived) database. A cracker who gains access to this database can then access every user account on a network, including the administrator's account. By default, Red Hat Enterprise Linux is released with all such services turned off. However, since administrators often find themselves forced to use these services, careful configuration is critical.

6. Threats To Workstation And Home PC

Workstations and home PCs may not be as prone to attack as networks or servers, but since they often contain sensitive data, such as credit card information, they are targeted by system crackers. Workstations can also be co-opted without the user's knowledge and used by attackers as "slave" machines in coordinated attacks. For these reasons, knowing the vulnerabilities of a workstation can save users the headache of reinstalling the operating system, or worse, recovering from data theft.Bad passwords are one of the easiest ways for an attacker to gain access to a system.

Although an administrator may have a fully secure and patched server that does not mean remote users are secure when accessing it. For instance, if the server offers Telnet or FTP services over a public network, an attacker can capture the plain text usernames and passwords as they pass over the network, and then use the account information to access the remote user's workstation. Even when using secure protocols, such as SSH, a remote user may be vulnerable to certain attacks if they do not keep their client applications updated. For instance, v.1 SSH clients are vulnerable to an X-forwarding attack from malicious SSH servers. Once connected to the server, the attacker can quietly capture any keystrokes and mouse Clicks made by the client over the network. This problem was fixed in the v.2 SSH protocol, but it is up to the user to keep track of what applications have such vulnerabilities and update them as necessary.

7. Vulnerabilities And Countermeasures

Following table consists of some common vulnerability in Linux and countermeasures that will harden the Linux security.

Table -1: Name of the Table

No	Vulnerability	Attack	Countermeasures
1	No separate partition for /boot, /, /home, /tmp, and /var/tmp	System crash and data loss	Create separate partition for /boot, /, /home, /tmp, and /var/tmp
2	Unnecessary software's	Software vulnerability Attack	Install Required software's only
3	Maliciously Altered Package	System instability ,System crash and data loss, data still	Install Signed Packages
4	No BIOS Password	Stealing/Changing Data Using a Bootable Linux CD	Give BIOS Password
5	Single User Mode access	Access as root user without password	Disable single user mode
6	Access to the GRUB Console	change its configuration or to gather information using the cat command.	Password Protecting GRUB
7	Access to Insecure Operating Systems	If it is a dual-boot system, an attacker can select an operating system at boot time (for example, DOS)	Password Protecting GRUB
8	Weak password, no password or default password	Cracking of weak passwords	1)Enforcing Stronger Passwords 2)Restricting Use of Previous Passwords 3)Locking User Accounts After Too Many Login Failures
9	No password Aging	Use of Cracked password over long period of time	Apply good password Aging Policy
10	root access to	1) Machine	1) Disallowing

	individual users	Misconfiguration 2) Running Insecure Services	Root Access 2) Disallow Remote Root Login 3) Disabling root access via any console device (tty)
11	OS fingerprinting	Get OS information like OS version etc.	Place login banner
12	Local log monitoring	Remove of log entries and log Files	Remote log monitoring
13	Insecure Services FTP , Telnet Transmit Usernames and Passwords Over a Network Unencrypted	1) Get user name and password. 2) Denial of Service Attacks (DoS)	1) Avoid these services and use behind the firewall 2) Use tcp_wrappers and xinetd 3) Use SSH
14	/etc/sysctl.conf configuration File vulnerability	1) SYN Attack 2) IP Source Routing 3) IP Spoofing 4) Broadcasts Request	Properly configure /etc/sysctl.conf Security Attribute

CONCLUSIONS

Thus the SELinux provides the more sophisticated security mechanism for securing Linux operating system. The baseline security configuration is almost usable for most environments, but some configuration is needed in most cases. The vital role in using SELinux is how you are configuring the SELinux security module. SELinux is a set of security policies/modules which are going to apply on the machine to improve the overall security of the machine. It is being included in most major Linux distributions, even though it might not be enabled by default. Installing or activating SELinux is pretty straight-forward, and no enforcement is being done until the user has checked the log files for possible problems and decides that the configuration is good enough.

Though Linux is much secure operating system various Linux Vulnerabilities and attacks occurs because of the weak configuration, Administration, unpatched services and insecure network architecture.

REFERENCES

[1] P. A. Loscocco, S. D. Smalley, P. A. Muckelbauer, R. C. Taylor, S. J. Turner, and J. F. Farrell. The Inevitabil-ity of Failure: The Flawed Assumption of Security in Modern Computing Environments. In 21st National Information Systems Security Conference, pages 303–314. NSA, 1998.R. Caves, Multinational Enterprise and Economic Analysis, Cambridge University Press, Cambridge, 1982. (book style)

[2] C. J. PeBenito, F. Mayer, and K. MacMillan. Refer-ence Policy for Security Enhanced Linux. In SELinux Symposium, 2006. H.H. Crokell, "Specialization and International Competitiveness," in Managing the Multinational Subsidiary, H. Etemad and L. S, Sulude (eds.), Croom-Helm, London, 1986. (book chapter style)

[3] R. Spencer, S. Smalley, P.Loscocco , M.Hibler, D. An-dersen, and J. Lepreau. The Flask Security Architec-ture: System Support for Diverse Security Policies. In The Eighth USENIX Security Symposium, pages 123–139, August 1999.

[4] Crispin Cowan, Steve Beattie, Calton Pu, PerryWagle, and Virgil Gligor. SubDomain: Parsimonious Server Security. In USENIX 14th Systems Administration Conference (LISA), New Orleans, LA, December 2000.

[5] Red_Hat_Enterprise_Linux-6-Security-Enhanced_Linux-en-US(Red Hat Engineering Content Services).