Journal of Information Engineering and Applications
ISSN 2224-5782 (print) ISSN 2225-0506 (online)
Vol 2, No.2, 2012

A Comparative Study of Security Mechanism differentiation between Windows 2000 and UNIX Operating Systems

Khalid Mahmood
ICIT, Gomal University, KPK, Pakistan
Khalid_icit@hotmail.com

Muhammad Javed
ICIT, Gomal University, KPK, Pakistan
Javed_gomal@yahoo.com

Muhammad Abid
ICIT, Gomal University, KPK, Pakistan
abid.gu@gmail.com

Bashir Ahmad
ICIT, Gomal University, KPK, Pakistan
bashahmad@gmail.com

Ihsan Ullah
ICIT, Gomal University, KPK, Pakistan
ihsanullah1974@yahoo.com

Allah Nawaz
Faculty of Pharmacy, Gomal University, Pakistan
nawaz_phd1@hotmail.com

Abstract:

"Security" is hard to formalize, hard to design (and design for), hard to implement, hard to verify, hard to configure, and hard to use. It is particularly hard to use on a platform such as Windows, which is evolving, security-wise, along with its representative user-base. The primary environment in which a typical Windows system exists has traditionally been hostile, especially after the advent of the Internet. While UNIX systems share the same environment today, their traditional environments were comparatively trusted: research labs and universities. Similarly, UNIX users have had backgrounds differing from Windows users.

Keywords: UNIX, Windows 2000, Security Mechanism, Operating System

1. Security in UNIX

1.1 Fundamental Concepts

UNIX (Tanenbaum, 2008) has been a multi-user system almost from the beginning. The user community of UNIX system consists of some number of registered users, each of whom has a unique User ID (UID). A UID is an integer between 0 to 65,535. Files are marked with the UID of their owner. By

default, the owner of a file is the person who created the file. Users can be organized into groups, which are also numbered with 16-bit integers called Group-ID done manually by System administrator. Originally a user could be in only one group, but now some version of UNIX allow user to be in more than one group. The security mechanism in UNIX is very simple, each process carried the UID and GID of its owner, when a file is created, its get the UID and GID of the creating process as well as set of permissions determined by the creating process. These permission specify the access of the owner, other members of the owner group, and the rest of users have to the file. Three categories of access are read, write and execute designated by the letter r, w and x respectively. Since there are three categories of users and 3 bits per category, 9 bits are sufficient to represent the access rights, such as

Binary	**Symbolic**	**Allowed file access**
111000000	Rwx------	Owner can read, write and execute
111111000	Rwxrwx---	Owner and group can read, write and Excute
000000111	------rwx	Only outsider have access(strange, But legal
000000000	---------	Denies access of any body.

The first two entries of the above table are very clear, allowing the owner and the owner group full access. The third entry is strange since it gives more access to rest of the world than owner and the last entry denies all access to all users. The user with UID 0 is special and is the called the Superuser or root. The supersuser has the power to read and write all files no matter who owns them and how they are protected. Process with UID 0 also has the ability to make a small number of protected system calls denied to ordinary users. Directories are files and have the same protection modes that ordinary file do except that x bits refer to search permission instead of execute permission. Thus a directory with mode rwxr-xr-x allows it owner to read, modify and search the directory, but allows other only to read and search it, but not add or remove files from it.

1.2 Security system calls in UNIX

There are only a small number of system calls relating to security. The most important are chmod, (it is used to change the protection mode), access, uid, gid, chown(change owner), setuid and setgid (set the GID).

1.3 Implementation of Security in UNIX

When the user logs in, the login program asks for the login name and a password. It hashes the password and then looks in the password file to check the user Login ID. The reason for hashing is to prevent the password from being stored in unencrypted form anywhere in the system. The login program then uses setuid and setgid to give itself the users ID and group ID. Then it opens the keyboard for standard input, the screen for standard output and the screen for standard error. Finally it executes the preferred shell, thus terminating itself. When any process attempts to open a file, the system first check s the protection bits in the file i-node against the caller effective UID and effective GID to see if the access is permitted. If so the file is opened and a file descriptor returned. If the file is not opened a -1 is returned. No checks are made on subsequent read and write calls. As a consequence, if the protection mode changes after a file is already open, the new mode will not affect process that, already have the file open. Security in the LINUX is essentially the same as in UNIX. All the UNIX security features are implemented and there is little else in this area.

2. Security in Windows (2000)

NT was designed to meet the U.S. Department of Defense C2 security requirements (wikipedia) the Orange book. This standard requires operating system to have certain properties in order to be classified as secure

enough for certain kinds of military work. Although windows 2000 was not specifically designed for C2 compliance, it inherits may securities properties from NT, which are

- Secure Login with anti-spoofing measures
- Discretionary access control
- Privileged access control
- Address space protection per process
- New pages must be zeroed before being mapped in
- Security auditing

Secure Login means that system administrator can require all users to have password in order to log in. Spoofing is when a malicious user writes a program that displays the login prompt or screen and then walks away from the computer in the hope that an innocent user will sit down and enter a name and password. The name and password are then written to disk and the user is told that login has failed. Windows 2000, Windows XP and Windows Vista prevents this attack by instructing user to hit CTRL-ALT-DEL (Secure Action Sequence) to log in. this key sequence is always captured by the keyboard driver, which then invokes a system program that puts up the genuine login screen. This procedure works because there is no way for user processes to disable CTRL-ALT-DEL processing in the keyboard driver. Discretionary access controls allow the owner of a file or other object to say who can use it and in what way. Privileged access control allows the system administrator (superuser) to override them when needed. Address space protection means that each has its won protected virtual address space not accessible by any unauthorized process. The new page must be zeroed means that when a stack grows, the pages mapped in are initialized to zero so processes cannot find any old information put there by the previous owner. Finally security auditing allows the administrator to produce a log of certain security related events.

2.1 Fundamental Concepts

Every windows 2000 user is identified by a SID (Security ID). SID is binary numbers with a short header followed by a long random component. Each SID is intended to be unique worldwide. Whenever a user starts a process, the process and its threads run under the user SID. Each process has an access token that specifies its SID and other properties. It is normally assigned at log in time, although process should call GetTokenInformation to acquire this information.

Header	Expiration	Groups	Default ACL	User SID	Group ID	Restricted ID	Privileges

Finally the privileges listed above, if any, give the process special powers, such the right to shut down the machine or access files to which would otherwise be denied. In effect, the privileges split up the power of superuser into several rights that can be assigned to processes individually. Another basic concept is the security descriptor. Every object has a security descriptor associated with it that tells who can perform which operations on it. The two main kinds of elements are Allow and Deny. An allow element specifies a SID and a bitmap that specifies which operation processes that with SID may perform on the object. A deny element work the same way, except a match means the caller may not perform the operation.

2.3 Security API Calls

Most of the Windows 2000 access control mechanism is based on security descriptors. The usual pattern is that when a process created an object, it provides a security descriptor as one of the parameters to the CreateProcess, createFile or other object creation call. This security descriptor is then attached to the object. Most of the Win32 API security calls relate to the management of security descriptor. To create a security descriptor, storage for it is first allocated and then initialized using InitializeSecurityDescriptor.

2.4 Implementation of Security

Security in Windows 2000 is implemented by a number of components. Logging is handled by winlogon and authentication is handled by lsass and msgina.dll. The result of successful login is a new shell with its associated access token. This process uses the SECURITY and SAM keys in the registry. When a user opens an object, the security manager checks to see if the caller has all the rights required. It performs this check by looking at the caller access token and the DACL associated with the object. As soon as it finds an entry that matches the caller's SID or one the callers groups the access found there is taken as definitive. If all the rights the caller needs are available, the open succeeds, otherwise it fails. After an object has been opened, a handle to it is returned to the caller. On subsequent calls, the only check that is made is whether the operation now being tried was in the set of operations requested at open time, to prevent a caller from opening a file for reading and then trying to write on it. Any log entries required by the ACL are made.

3. Comparison of Security Protection Mechanism b/w UNIX and Windows (wikipedia, 2009)

	Windows	Linux
Malware	According to Kaspersky Lab, more than 11,000 malware programs for Windows were discovered in the second half of 2005. Botnets - networks of infected computers controlled by malicious persons - with more than one million computers have been witnessed. Users are advised to install and run anti-malware programs.	More than 800 pieces of Linux malware have been discovered. Some malware has propagated through the Internet.
Open vs. Closed	Claims its platform is more secure because of a comprehensive approach to security using the Security Development Lifecycle. However, due to the nature of closed source, only company programmers can fix bugs.	Claims its platform is more secure because all of its code is reviewed by so many people that bugs are detected
Response speed	Claims closed source offers a faster and more effective response to security issues, though critical bug fixes are only released once a month after extensive programming and testing and certain bugs have been known to go unpatched for months.	Bugs can be fixed and rolled out within a day of being reported, though usually it takes a few weeks before the patch is available on all distributions.
User Accounts	In Windows Vista, all logged in sessions (even for those of "administrator" users) run with standard user permissions, preventing malicious programs from gaining total control of the system. Prior versions of Windows would assign administrator status to the first user account created during the setup process. The majority of users did not change to an	Users typically run as limited accounts, having created both administrator and user accounts during install, preventing malicious programs from gaining total control of the system.

	account type with fewer rights, meaning that malicious programs would have full control over the system.	

4. Conclusion

Linux and Windows differ in many aspects. First of all, the Linux GUI is optional while the Windows GUI is an integral component of the OS; speed, efficiency and reliability are all increased by running a server instance of Linux without a GUI, something that server versions of Windows can not do. The detached nature of the Linux GUI makes remote control and remote administration of a Linux computer simpler and more natural than a Windows Computer. Secondly the command prompts of these operating systems are quite different. In general, the command interpreters in the Windows 9x series are very similar to each other and the NT class versions of Windows (NT, 2000, XP) also have similar command interpreters. There are, however differences between a Windows 9x command interpreter and one in an NT class flavor of Windows. Linux, like all versions of UNIX, supports multiple command interpreters, but it usually uses one called BASH (Bourne Again Shell). Others are the Korn shell, the Bourne shell, ash, and the C shell (pun, no doubt, intended). The costs are amazingly different. While you have to pay some hundred dollars for a new version of Windows, you can simply go and download Linux. As it comes from the nature of Linux, there are no manuals or simple installers for the free version, however. You really have to know what you are doing while using this free package. There are also some easy automated packages of Linux for low prices, as well. The security issues with Windows are the biggest cons of Microsoft. Most of the malicious files, spyware, adware programs deal with Windows. You generally do not deal with these kinds of unwanted circumstances unless you are working with Windows. The user-id and password protection for Windows can also be easily bypassed, whereas Linux offers a strong protection. After mentioning some of the different aspects of these operating systems, it can be said that all Linux needs to compete with Windows is some user friendly interface and a strong company support which can provide the users with technical information and user manuals

References:

Modern Operating System, 2nd Edition, Andrew S. Tanenbaum

Difference b/w Unix and Windows [online] Available :http://www.wikipedia.com

Security mechanism difference b/w windows and UNIX [online] Available: http://www.google.com.pk