สิทธิพิเศษ (คอมพิวเตอร์)

ในการคำนวณ**สิทธิพิเศษ**ถูกกำหนดให้เป็นผู้มอบอำนาจเพื่อทำหน้าที่ที่เกี่ยวข้องกับความปลอดภัยบนระบบคอมพิวเตอร์ [1] สิทธิ์อนุญาตให้ผู้ใช้ดำเนินการกับผลด้านความปลอดภัย ตัวอย่างของสิทธิพิเศษต่างๆ ได้แก่ ความสามารถในการสร้างผู้ใช้ใหม่ ติดตั้งซอฟต์แวร์ หรือเปลี่ยนฟังก์ชันเคอร์เนล

ผู้ใช้ที่ได้รับมอบหมายระดับการควบคุมพิเศษจะเรียกว่ามีสิทธิพิเศษ ผู้ใช้ที่ไม่มีสิทธิ์ส่วนใหญ่จะถูกกำหนดให้เป็นผู้ใช้ที่ไม่มีสิทธิ์ ผู้ใช้ปกติ หรือผู้ใช้ทั่วไป

ทฤษฎี

Privileges can either be automatic, granted, or applied for.

An automatic privilege exists when there is no requirement to have permission to perform an action. For example, on systems where people are required to log into a system to use it, logging out will not require a privilege. Systems that do not implement file protection - such as MS-DOS - essentially give unlimited privilege to perform any action on a file.

A granted privilege exists as a result of presenting some credential to the privilege granting authority. This is usually accomplished by logging on to a system with a username and password, and if the username and password supplied are correct, the user is granted additional privileges.

A privilege is applied for by either an executed program issuing a request for advanced privileges, or by running some program to apply for the additional privileges. An example of a user applying for additional privileges is provided by the sudo command to run a command as the root user, or by the Kerberos authentication system.

Modern processor architectures have multiple CPU modes that allows the OS to run at different privilege levels. Some processors have two levels (such as *user* and *supervisor*); i386+ processors have four levels (#0 with the most, #3 with the least privileges). Tasks are tagged with a privilege level. Resources (segments, pages, ports, etc.) and the privileged instructions are tagged with a demanded privilege level. When a task tries to use a resource, or execute a privileged instruction, the processor determines whether it has the permission (if not, a "protection fault" interrupt is generated). This prevents user tasks from damaging the OS or each other.

In computer programming, exceptions related to privileged instruction violations may be caused when an array has been accessed out of bounds or an invalid pointer has been dereferenced when the invalid memory location referenced is a privileged location, such as one controlling device input/output. This is particularly more likely to occur in programming languages such as C, which use pointer arithmetic or do not check array bounds automatically.

Unix

On Unix-like systems, the superuser (commonly known as 'root') owns all the privileges. Ordinary users are granted only enough permissions to accomplish their most common tasks. UNIX systems have built-in security features. Most users cannot set up a new user account nor do other administrative procedures. The user “root” is a special user, something called super-user, which can do anything at all on the system. This high degree power is necessary to fully administer a UNIX system, but it also allows its user to make a mistake and cause system problems.

Unprivileged users usually cannot:

- Adjust kernel options.
- Modify system files, or files of other users.
- Change the ownership of any files.
- Change the runlevel (on systems with System V-style initialization).
- Change the file mode of any files.
- Adjust ulimits or disk quotas.
- Start, stop and remove daemons.
- Signal processes of other users.

- Create device nodes.
- Create or remove users or groups.
- Mount or unmount volumes, although it is becoming common to allow regular users to mount and unmount removable media, such as Compact Discs. This is typically accomplished via FUSE.
- Execute the contents of any `sbin/` directory, although it is becoming common to simply restrict the behavior of such programs when executed by regular users.
- Bind ports below 1024.

Windows NT

On Windows NT-based systems, privileges are delegated in varying degrees. These delegations can be defined using the Local Security Policy Manager (SECPOL.MSC). The following is an abbreviated list of the default assignments:

- 'NT AUTHORITY\System' is the closest equivalent to the Superuser on Unix-like systems. It has many of the privileges of a classic Unix superuser, such as being a trustee on every file created
- 'Administrator' is one of the closest equivalents to the Superuser on Unix-like systems. However, this user cannot override as many of the operating system's protections as the Superuser can.
- Members of the 'Administrators' group have privileges almost equal to 'Administrator'.
- Members of the 'Power Users' group have the ability to install programs and backup the system.
- Members of the 'Users' group are the equivalent to unprivileged users on Unix-like systems.

Windows defines a number of administrative privileges[2] that can be assigned individually to users and/or groups. An account (user) holds only the privileges granted to it, either directly or indirectly through group memberships. Upon installation a number of groups and accounts are created and privileges are granted to them. However, these grants can be changed at a later time or though a group policy. Unlike Linux, no privileges are implicitly or permanently granted to a specific account.

Some administrative privileges (e.g. taking ownership of or restoring arbitrary files) are so powerful that if used with malicious intent they could allow the entire system to be compromised. With user account control (on by default since Windows Vista) Windows will strip the user token of these privileges at login. Thus, if a user logs in with an account with

broad system privileges, he/she will still not be *running* with these system privileges. Whenever the user wants to perform administrative actions requiring any of the system privileges he/she will have to do this from an *elevated* process. When launching an *elevated* process, the user is made aware that his/her administrative privileges are being asserted through a prompt requiring his/her consent. Not holding privileges until actually required is in keeping with the Principle of least privilege.

Elevated processes will run with the full privileges of the *user*, not the full privileges of the *system*. Even so, the privileges of the user may still be more than what is required for that particular process, thus not completely least privilege.

The DOS-based Windows ME, Windows 98, Windows 95, and previous versions of non-NT Windows only operated on the FAT filesystem and did not support filesystem permissions.,[3] and therefore privileges are effectively defeated on Windows NT-based systems that do not use the NTFS file system.

Nomenclature

The names used in the Windows source code end in either "Privilege" or "LogonRight". This has led to some confusion about what the full set of all these "Rights" and "Privileges" should be called.

ปัจจุบัน Microsoft ใช้คำว่า "สิทธิ์ผู้ใช้" [4] ในอดีต Microsoft ก็เคยใช้คำศัพท์อื่นๆ เช่น "Privilege Rights" [5] , "logon user rights" [6] and "NT-Rights".[7]

ดูสิ่งนี้ด้วย

- Principle of least privilege
- Superuser
- File system permissions
- Kernel (computer science)

อ้างอิง

1. *"อภิธานศัพท์" (https://csrc.nist.gov/glossary/term/privileged-user) . ศูนย์วิทยบริการรักษาความปลอดภัยคอมพิวเตอร์ NIST สืบค้นเมื่อ12 กุมภาพันธ์ 2019 .*
2. *"ค่าคงที่ของสิทธิพิเศษ" (http://msdn.microsoft.com/en-us/library/bb530716.aspx) . ไมโครซอฟต์.*

3. *"How Permissions Work" (https://technet.microsoft.com/en-us/library/cc783530(WS.10).aspx)* . *Microsoft. "You can set permissions at the file level only if the files are stored on an NTFS volume."*
4. *"User Rights" (https://technet.microsoft.com/en-us/library/dd349804(v=ws.10).aspx)* . *Microsoft TechNet Library. "User rights include logon rights and privileges."*
5. *"Privilege Rights" (http://msdn.microsoft.com/en-us/library/cc232779(prot.20).aspx)* . *Microsoft MSDN Library.*
6. *"How to set logon user rights by using the NTRights utility" (http://support.microsoft.com/kb/315276)* . *Microsoft Support. "The following is a list of logon user rights [...] SeInteractiveLogonRight [...] SeDebugPrivilege"*
7. *"How to set logon user rights by using the NTRights utility" (http://support.microsoft.com/kb/315276)* . *Microsoft Support. "NTRights.Exe [...] Grants/Revokes NT-Rights [...] valid NTRights are: SeCreateTokenPrivilege"*

แก้ไขล่าสุดเมื่อ14 วันก่อนโดยGuy Harris