Privileged instruction | Article about privileged instruction by The Free Dictionary

https://encyclopedia2.thefreedictionary.com/privileged+instruction

privileged instruction

privileged instruction ['priv·ə·lijd in'strək·shən]

(computer science)

A class of instructions, usually including storage protection setting, interrupt handling, timer control, input/output, and special processor status-setting instructions, that can be executed only when the computer is in a special privileged mode that is generally available to an operating or executive system, but not to user programs.

McGraw-Hill Dictionary of Scientific & Technical Terms, 6E, Copyright © 2003 by The McGraw-Hill Companies, Inc.

privileged instruction

A **machine code** instruction that may only be executed when the processor is running in **supervisor mode**. Privileged instructions include operations such as I/O and **memory management**.

This article is provided by FOLDOC - Free Online Dictionary of Computing ()

WIKIPEDIA

Privilege (computing)

In computing, **privilege** is defined as the delegation of authority to perform security-relevant functions on a computer system.[1] A privilege allows a user to perform an action with security consequences. Examples of various privileges include the ability to create a new user, install software, or change kernel functions.

Users who have been delegated extra levels of control are called privileged. Users who lack most privileges are defined as unprivileged, regular, or normal users.

Contents

Theory

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 superuser (*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 (root) 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, 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.

Microsoft currently uses the term "user rights".[4] In the past some other terms have also been used by Microsoft, such as "privilege rights"[5] , "logon user rights"[6] and "nt-rights".[7]

See also

- File-system permissions
- Kernel (operating system)
- Principle of least privilege
- Superuser
- Privilege escalation

References

1. "Glossary" (https://web.archive.org/web/20190213005620/https://csrc.nist.gov/glossary/term/privileged-user). *CSRC*. NIST. Archived from the original (https://csrc.nist.gov/glossary/term/privileged-user) on 13 February 2019. Retrieved 12 February 2019.
2. "Privilege constants" (http://msdn.microsoft.com/en-us/library/bb530716.aspx). Microsoft.

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. "Userrights 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.
7. "How to set logon user rights by using the ntrights utility" (http://support.microsoft.com/kb/315276). Microsoft support.

DOCKET ALARM

Explore Litigation Insights

Docket Alarm provides insights to develop a more informed litigation strategy and the peace of mind of knowing you're on top of things.

Real-Time Litigation Alerts

Keep your litigation team up-to-date with **real-time alerts** and advanced team management tools built for the enterprise, all while greatly reducing PACER spend.

Our comprehensive service means we can handle Federal, State, and Administrative courts across the country.

Advanced Docket Research

With over 230 million records, Docket Alarm's cloud-native docket research platform finds what other services can't. Coverage includes Federal, State, plus PTAB, TTAB, ITC and NLRB decisions, all in one place.

Identify arguments that have been successful in the past with full text, pinpoint searching. Link to case law cited within any court document via Fastcase.

Analytics At Your Fingertips

Learn what happened the last time a particular judge, opposing counsel or company faced cases similar to yours.

Advanced out-of-the-box PTAB and TTAB analytics are always at your fingertips.

API

Docket Alarm offers a powerful API (application programming interface) to developers that want to integrate case filings into their apps.

LAW FIRMS
Build custom dashboards for your attorneys and clients with live data direct from the court.

Automate many repetitive legal tasks like conflict checks, document management, and marketing.

FINANCIAL INSTITUTIONS
Litigation and bankruptcy checks for companies and debtors.

E-DISCOVERY AND LEGAL VENDORS
Sync your system to PACER to automate legal marketing.