Operating system and file security v1

Note Title

① Processes and their permissions

Computers appear to do many things at the same time, because they are constantly switching between processes — typically, a switch occurs every few milliseconds.

The list of processes can be viewed via Task Manager (on Windows) or Activity Manager (OS/X).

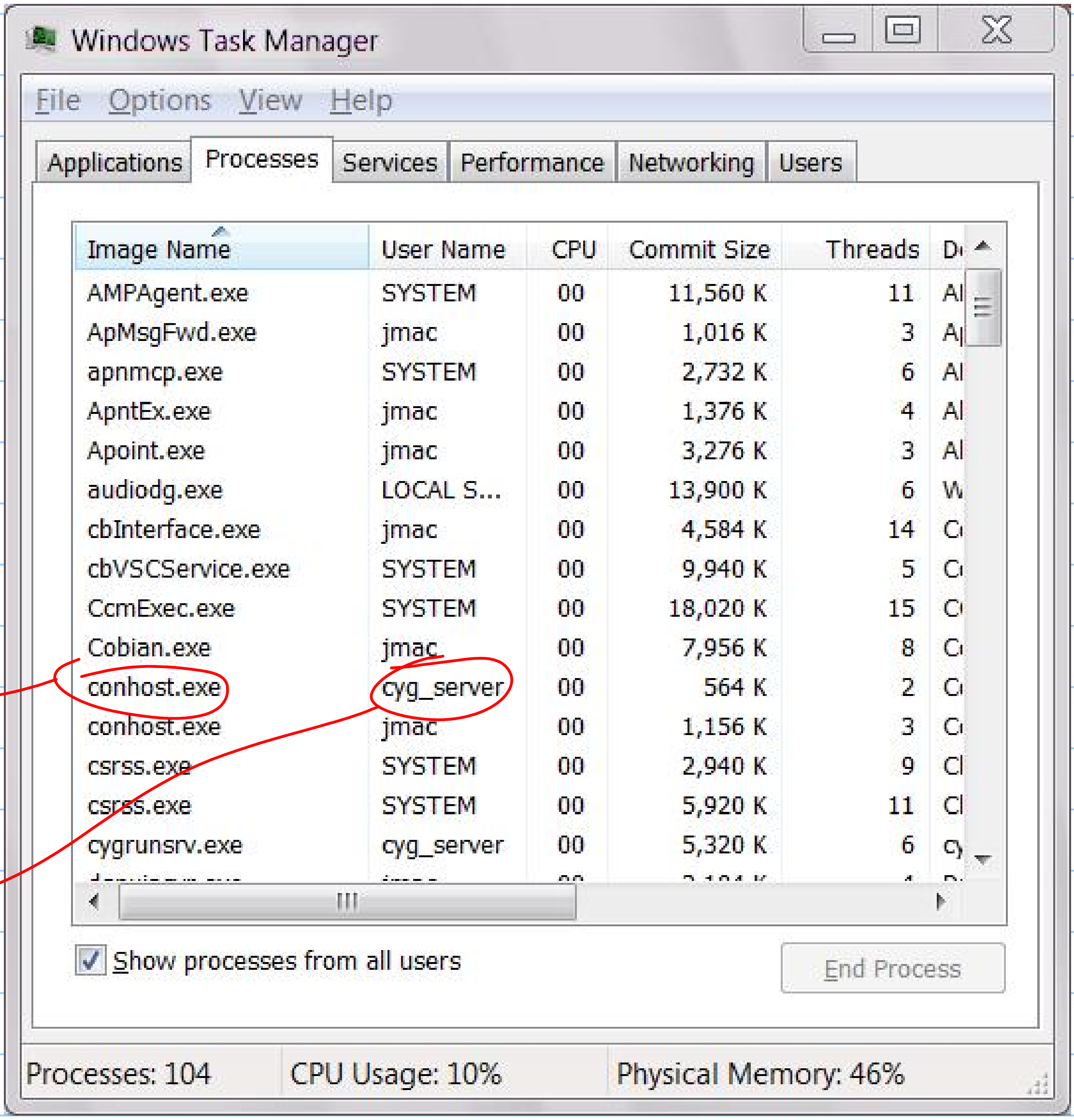

Image Name	User Name	CPU	Commit Size	Threads	D
AMPAgent.exe	SYSTEM	00	11,560 K	11	Al
ApMsgFwd.exe	jmac	00	1,016 K	3	Ap
apnmcp.exe	SYSTEM	00	2,732 K	6	Al
ApntEx.exe	jmac	00	1,376 K	4	Al
Apoint.exe	jmac	00	3,276 K	3	Al
audiodg.exe	LOCAL S...	00	13,900 K	6	W
cbInterface.exe	jmac	00	4,584 K	14	C
cbVSCService.exe	SYSTEM	00	9,940 K	5	C
CcmExec.exe	SYSTEM	00	18,020 K	15	C
Cobian.exe	jmac	00	7,956 K	8	C
conhost.exe	cyg_server	00	564 K	2	C
conhost.exe	jmac	00	1,156 K	3	C
csrss.exe	SYSTEM	00	2,940 K	9	Cl
csrss.exe	SYSTEM	00	5,920 K	11	Cl
cygrunsrv.exe	cyg_server	00	5,320 K	6	cy

☑ Show processes from all users

Processes: 104 CPU Usage: 10% Physical Memory: 46%

name of file being executed

name of user executing process

Roughly speaking, a process is a separate program, executing sequences of instructions from an executable file. (See left column above). Some applications correspond to a single process (e.g. excel) but some apps use many processes (e.g. MS Word). Many processes have no corresponding app. [demo]

The operating system provides 2 key types of security to each process: memory isolation, and user privileges.

Memory isolation

Each process is isolated from the others by the OS.

Specifically, the OS decides which parts of the computer's memory (RAM) can be accessed by each process.

One process cannot alter the memory assigned to another process.

User priveleges

The OS keeps a list of user accounts, which may or may not correspond to human users (e.g. jmac, cyg-server, SYSTEM in the 2nd column above).

Each user has certain privileges to access files, network etc, as described later.

Each process is executed by some user (see 2nd column above again).

The process can only do things that the corresponding user is permitted to do.

[Demo: MS Word run as different users. Show differing abilities to save as].

Most security exploits center around hijacking a process that is run as 'root' or 'administrator'

↑ ↑

on linux/OSX on windows

The Principle of Least Privilege

This is a general principle of Computer security, which is particularly applicable to process privileges. The principle states that an entity should have the least amount of privileges necessary to perform its function.

e.g. a process that needs to *read* the file "wombat.docx" but does not need to *modify* that file, should have readonly access to that file.

② File permissions and security

or folder – means the same thing

Associated with every file and directory on a computer is a set of permissions, that can include:

or 'modify' – means the same thing

- read : can examine the content
- write : can write to the file – i.e. change the content
- execute : can run the file as a program
- list : can list the contents of a folder
- create : can create new files in a folder
- delete : can delete files in a folder

Each file and directory has an owner (a user or group of users) who can alter these permissions.

These permissions can typically be set differently for different users.

[demo: show read, write, execute etc... in action. show how it applies to different users. Also show 'read only' - useful even for owner.]

Typical settings include:

- completely private: owner can read, but can't write. no-one else can read or write. e.g. one of your private keys; personal files

- public: owner can read or write, anyone else can read (aka "world-readable") e.g. one of your public keys; one of your web pages

Deny by default

This is a general security principle: access should always be denied unless permission has been specifically granted.

e.g. when a new file is created, the OS will typically deny read and write access to all but the owner.

[demo: create a new file, and see what the permissions are]

Groups:

Most operating systems make it easier to specify permissions by defining groups of users. File permissions can be given to (or denied to) a given group.

e.g. (a) We could create a group "teaching-assistants" and give read permission to that group for a directory containing student homework.

(b) The 'administrators' group in Windows has permission to change most settings on a computer.

[demo: see what groups exist on current machine.]

Access control lists:

An Access control list (ACL) for a given file is just a list of all the groups and users that have been granted or denied permissions for the file.

[demo: ACLs in Windows]

All modern operating systems provide ACLs as a way of managing permissions, but they are not the only way.

POSIX / UNIX permissions

ACLs can get very complicated and confusing.
[demo: 'effective permissions' on Windows]

A much simpler but highly effective alternative, known as 'UNIX' or 'POSIX' permissions is also available on most operating systems. We will examine this further in our lab.