Poly-Instantiated Directories and SE Linux

Russell Coker <russell@coker.com.au>
Internet and Security Consulting

- □ The problem space for non-MLS systems
- □ MLS requirements
- □ Linux implementation
- □ How well the problem is solved

The problem space for non-MLS systems

- Traditionally /tmp and /var/tmp are used for temporary storage by all users and daemons

- Many programs use fixed or predictable file names permitting race-condition attacks (including DOS attacks)

- Sometimes the file name may contain sensitive information, it may be created without the user’s knowledge

- Need to solve this without re-writing tens of thousands of programs and re-training millions of users

Examples

- strace -ff creates easily predictable file names
- Users act predictably
- Buggy daemons such as rlogind
- Buggy applications such as unzip CAN-2005-2475 and bzip2 CAN-2005-0953 can grant an attacker access to arbitrary data due to sym-link race conditions
- Buggy applications such as fbi CVE-2006-1695 can have a DOS attack if a predictabl directory exists
- Programs perform unknown operations on behalf of users (EG editors creating files under /tmp or /var/tmp)

Specific Attack Scenarios

- Attack by user on user
- Attack by user on daemon
- Attack by non-root daemon on user
- Attack by root daemon on user

Previous attempts

- Restrictions on creating links - OpenWall

- Hiding file names, only works for the case where file names are secret, not for boolea file names

MLS requirements

- □ Multiple instances of home directory for each sensitivity level
- □ Multiple instances of shared directories with sensitive file names being the main motivation
- □ MLS itself solves the confidentiality issues related to reading files and the SE Linux domain-type model solves most integrity issues related to writing and reading files (ar mitigates the rest), so sensitive file names is the remaining problem
- □ There are situations where users who have the same SE Linux role and MLS level need to be prevented from seeing each other’s data

Linux implementation

- New systemcall unshare() to create private name-space for filesystems (among other things) - can be called from PAM module to work with unmodified programs
- Directory such as /tmp/.inst/tmp.inst-rjc-rjc is created and bind mounted to /tmp
- /proc/self/mounts shows the filesystems mounted for a process, /proc/mounts links to /proc/self/mounts
- PAM setting

session required pam_namespace.so

- Option unmnt_remnt for su and comparable programs (probably suexec, maybe MTA local delivery)

Shared-subtrees

- Allow autofs and sys-admin mount commands to work

```
mount --make-shared /
mount --bind /tmp /tmp
mount --make-private /tmp
```

- Only works on mount points, bind mount of /tmp needed for /tmp in root FS

- If PI directories are not excluded from the shared name space then things go horribly wrong

How well the problem is solved

- Non-root daemons started via runuser will have PI
- User processes from regular login and cron jobs have PI
- Support for excluding some users from PI, to prevent them from attacking PI users an daemons all directories are under /tmp/.inst which is a mode 000 directory
- Adds significant integrity and confidentiality benefits both with and without SE Linux
- On SE Linux systems there is an option of instantiating based on context, UID, or both

Further Work

- All initial goals met - new design goals after paper was written

- Daemons such as Apache that change UID after being started are not run with PI, need wrapper for this

- Need suexec support, support for local MTA delivery, and possibly other support for system processes acting on behalf of users

- Probably need to make more daemons support PAM session, suexec and postfix/loca are two good possibilities

Q/A

- #selinux on irc.freenode.net
- http://www.nsa.gov/selinux/   Official SE Linux web site
- http://www.coker.com.au/selinux/   My SE Linux web pages

- Russell Coker <russell@coker.com.au>

- Internet and Security Consulting