Robert Lemos, News.com, 2004.11.08:

“Virus writers elude Microsoft’s bounty hunt

“Virus writers have a price on their heads—but it’s done little to discourage them.

“In the year since Microsoft kicked off its Anti-Virus Reward Program, it has tallied only a single success. The program has offered US$1 million to informants who help close official investigations into four major viruses and worms, and has another US$4 million earmarked for future rewards, but the deluge of online threats has continued to swell. . . .

"German authorities arrested a teenager in May after Microsoft tipped them off with details about the alleged Sasser author it had received from . . . a friend of the suspected author. . . .

"The Sasser worm, which started spreading on May 1, has infected an estimated 500,000 to 1 million systems . . . If the alleged author of the worm, Sven Jaschan, is convicted of criminal charges, Microsoft will be on the hook to pay out the bounty. . . .

"A security company has hired Jaschan, pending his conviction."

2004.11.15: Guest lecture
by Jon Solworth, Director,
Kernel Security and Networking Lab,
CS.

2004.11.17: Midterm 2,
focusing on setuid and related topics.

Assignment due 2004.11.22: read
textbook Chapter 4.

The file-rewriting problem

Joe runs setuid program `chsh`
that reads `/etc/passwd`,
writes modified `/etc/passwd`.

`chsh` locks `/etc/passwd`
while rewriting it; other programs
wait until the file is complete.

Security problem: Joe sends a signal
to `chsh`, terminating it,
before the file is complete.
The rest of the file has been destroyed.

Fix: `chsh` can avoid signals.

Another way to hurt rewriting

Joe can fill up the disk,
so `chsh` has no space
to rewrite `/etc/passwd`
after truncating it.

Joe can attack any program,
not just setuid programs,
in this way.

Fix: Put `/etc/passwd` on one disk.
Put `/home/joe` on another disk.
(Or set a "quota" on Joe's files.)
Carefully review all the files
outside `/home/joe` that might
expand under Joe's influence.

A better way to rewrite files

`chsh` can write `/etc/passwd.new`
and then, when it's complete,
`rename("/etc/passwd.new",`
`"/etc/passwd").`

If `rename` succeeds,
`/etc/passwd` has the new contents.
If `rename` fails,
`/etc/passwd` is unchanged.
Partial success is impossible;
`rename` is **atomic**.

Details: use a separate lock file;
check carefully for errors. Can even
handle power outages: use `fsync`
after writing, and use a careful filesystem.

Recap

Easy but lets Joe corrupt file:
don't worry; be happy;
write directly to `/etc/passwd`.

Slightly more difficult,
doesn't let Joe corrupt file:
write `/etc/passwd.new`; rename.

Substantially more difficult,
doesn't let Joe corrupt file:
write directly to `/etc/passwd`,
after eliminating signals and full disks.

Guess what most people do?

Sendmail bug fixed 1995.09.16:
"... destroying the alias database file by setting resource limits low."

More bugs discovered later.

(Did programmer consider `rename`?
Silly database design:
the database was actually two files
that had to be updated together.
Can't do simultaneous `rename`
of two files.)

Eventual fix, 2000.03.01:
Nobody other than sysadmin
can touch the database file.

What else affects a setuid program?

See `execve` manual page.

BSD: "File descriptors open in the calling process image remain open ... Signals set to be ignored in the calling process are set to be ignored ... Blocked signals remain blocked ... The new process also inherits the following attributes from the calling process: process ID, parent process ID, process group ID, access groups, working directory, root directory, control terminal, resource usages, interval timers, resource limits, file mode mask, signal mask."

Attacker blocking signals

Each process has, in system data,
signal mask and **signal actions**.

Mask says which signals are blocked.
Actions say which signals are ignored;
which signals call functions;
the addresses of those functions.

(If a signal arrives and is blocked,
it is saved until the signal is un-blocked.
If a signal arrives and is ignored,
it is discarded.)

Blocked and ignored signals
are preserved by `execve`.

Last time: Setuid program
wasn't expecting a signal.
Joe sends it one.

Opposite problem: Setuid program
sends itself a signal;
needs signal to make progress.
Joe blocks the signal
before running the program.

Fix: Program un-blocks signal
and un-ignores signal, using
`sigprocmask` and `sigaction`.

Attacker blocking permission bits

Each process has, in system data,
umask ("file mode mask").

Typical umask: `022`.
Another typical umask: `077`.

Any permission bit in umask
is removed from new files.

e.g. `open("foo",O_CREAT,0666)`
creates `foo` with permissions
`0644` if umask is `022`;
or `0600` if umask is `077`.