John Markoff, New York Times, 2004.10.25:

“New I.B.M. report will warn of computer security threats

“I.B.M. plans to begin releasing on Monday a monthly report of threats to computer networks in an effort to establish an indicator similar to the federal government’s Homeland Security Advisory System.

“The report, to be named the Global Business Security Index, is intended to give computing managers early warning of a range of computer vulnerabilities like attacks by malicious hackers, automated

softwares, viruses and worms, as well as to gauge the impact of political upheavals and natural disasters. . . .

"The index will be released on I.B.M.'s Web site and will be part of a broader service known as the I.B.M. Security Threats and Attack Trends, or STAT, report, which the company offers customers at a cost of about $10,000 a year. . . .

"Both the I.B.M. executives and other security experts said that they were seeing more sophisticated attacks and that the culture of the computer underground was shifting from bored teenagers to criminals attempting to steal information or money."

Course grade:
60% homework.
10% midterm 1.
10% midterm 2, probably 17 November.
20% final.

Need 85% for A, 75% for B, etc.

Each exam question is
scored on a 10-point scale.
5 questions on each midterm,
10 questions on final.

6% first security hole;
6% second security hole;
...
6% tenth security hole.
2% first question on midterm 1;
2% second question on midterm 1;
...
2% fifth question on midterm 1.
2% first question on midterm 2;
2% second question on midterm 2;
...
2% fifth question on midterm 2.
2% first question on final;
2% second question on final;
...
2% tenth question on final.

Before calling `execve`,
most setuid programs call

```
setuid(getuid());
```

to set uid to real uid.

Does `setuid(getuid())`
really give up all extra powers
obtained by a setuid program?
Not necessarily!

1. Can undo `setuid(getuid())`
in some situations.

2. Process may have read secrets
while it had `root` access.
Relies on `execve` to wipe memory.

After `setuid(getuid())`,
some systems allow user to
extract secrets from memory
by attaching with `gdb`.

Easy fix: attaching never works
if program has permissions 4711.
But most setuid programs are 4755.

(Some UNIX kernel bugs have
allowed attaching even before
`setuid(getuid())`; see, e.g.,
Linux kernel bug fixed 2001.02.)

All permission bits:

4000: setuid to file's owner.

2000: setgid to file's group.

1000: sticky; restricts directories.

0400: readable by owner.

0200: writable by owner.

0100: executable by owner.

0040: readable by group.

0020: writable by group.

0010: executable by group.

0004: readable by other users.

0002: writable by other users.

0001: executable by other users.

Groups are analogous to owners
but somewhat more complicated.

3. Process may have acquired
other privileges while `root`:
e.g., it may have opened a
`root` file to read secrets
or to write something.

`setuid()` doesn't close file.
`execve()` doesn't close file.

`/home/joe/evil` can read file
if it was opened for reading:
`read()` syscall doesn't re-check uid.
Similarly, can write file
if it was opened for writing.

Sendmail bug, fixed 2000.03.01:
failed to close various files
before `execv(argv[0],argv)`.

File descriptors

Each process has an array of
file descriptors inside system data.

Some important components of each fd:

- "readable": 1 if the file is open for reading, 0 otherwise;
- "writable": 1 if the file is open for writing, 0 otherwise;
- "type": 1 for disk file, 2 for network socket, 3 for pipe, etc.
- "inode number": where the file is located on disk, if it's a disk file;
- "position": location in file of next byte to read or write; ...

Details: `/usr/include/sys/file.h`, `/usr/include/sys/socket.h`, et al.