Security

CS 256/456

Dept. of Computer Science, University of Rochester

The Security Environment

Security goals:
- Data confidentiality
- Data integrity
- System availability

Threats of intruders or adversaries:
- Exposing data
- Tampering with data
- Denial of service attacks

We focus on OS-related security issues.

User Authentication

```
LOGIN: ken
PASSWORD: FooBar
SUCCESSFUL LOGIN
```

(a)

```
LOGIN: carol
INVALID LOGIN NAME
LOGIN:
```

(b)

```
LOGIN: carol
PASSWORD: Idunno
INVALID LOGIN
LOGIN:
```

(c)

(a) A successful login
(b) Login rejected after name entered
(c) Login rejected after name and password typed

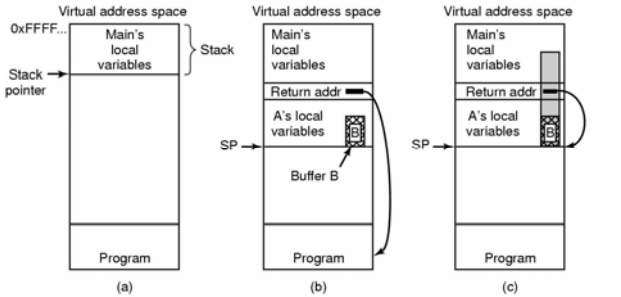
User Authentication

- UNIX user passwords are mapped using a one-way function "e()"; and then stored in a globally readable file "/etc/passwd"
  - Bobbie, e(Dog)
  - Tony, e(6%%TaeFF)
  - … …
- Attacks:
  - used a precomputed common password list
  - exhaustive attack
- Countermeasure?
  - salt
  - Bobbie, 4238, e(Dog4238)
  - Tony, 2918, e(6%%TaeFF2918)

Login Spoofing

- Login spoofing
  - A program running by the attacker displays a login screen (like the real one)
  - After a legitimate user types in username and password, it records those, kills itself, and a real login screen is shown
  - The user thinks she typed in a wrong password and tries again, which works
- Countermeasure?
  - Start each login session with a non-user-catchable key combination "Ctl-Alt-Delete"

Buffer Overflow



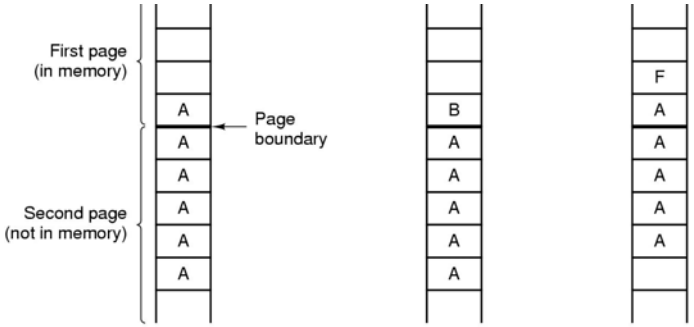
(a) Situation when main program is running
(b) After program *A* called
(c) Buffer overflow shown in gray

Countermeasures:

- boundary checks, non-executable stack/data segment, …

The TENEX Password Problem

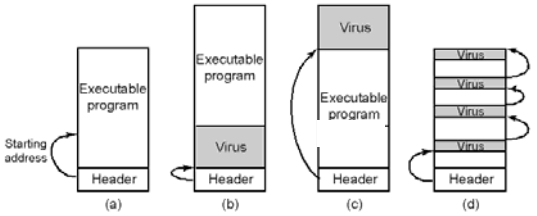
Files are accessed with passwords. At each access, the password is checked byte-by-byte and an error is returned as soon as a byte is mismatched.



Virus

- Virus
  - program can reproduce itself
    - e.g., when invoked, traverse the file system and attach it to randomly selected executables
  - additionally, do harm
    - denial of service by using all available system resources
    - permanently damage data or hardware
- "Good" virus:
  - quickly spreading virus
  - difficult to detect
  - hard to get rid of

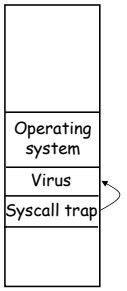
Infecting An Executable (Trojan Horses)



(a) An executable program
(b) With a virus at the front
(c) With the virus at the end
(d) With a virus spread over free space within program

Memory Resident Viruses

- Virus resides in memory; intercepting system calls
- Where to put the virus?
- How to load virus there in the first place?



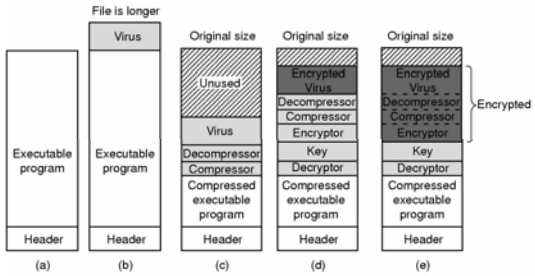
How Viruses Spread

- Try to infect programs on
  - networks: exploiting buffer overflow errors in network server daemons
  - floppy drives
- Attach to innocent looking email
  - when it runs, use mailing list to replicate

Antivirus Techniques

- Signature scanning
  - maintain a database of patterns of common viruses
  - scan disk files for these patterns
- Size checkers
  - keep a record on the size of disk files and scan them periodically for any size changes
  - apply on readonly executables.

Anti-Antivirus Techniques

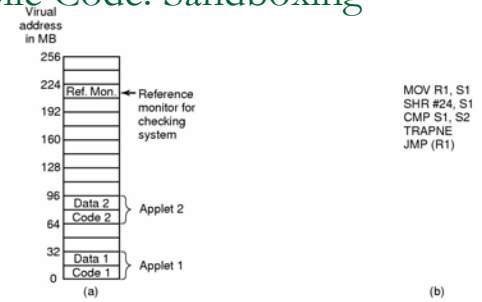


(a) A program
(b) Infected program
(c) Compressed infected program
(d) Encrypted virus
(e) Compressed virus with encrypted compression code

More Antivirus Techniques

- Integrity checkers
  - similar to size checkers, but this time we compute a checksum for file and store them somewhere; we periodically check all files to see whether the checksum still matches
- Behavioral checkers (memory-resident anti-virus program)
  - intercept system calls and detect suspicious activities: overwriting the boot sector, overwriting executables, …

Mobile Code: Sandboxing



Memory divided into 16-MB sandboxes
- applet can't jump out of its code sandbox
- applet can't access data out of its data sandbox

Limited system call capabilities
- all system calls must go through a reference monitor

Enforcement: binary rewriting or interpretation

Disclaimer

- Parts of the lecture slides contain original work of Abraham Silberschatz, Peter B. Galvin, Greg Gagne, Andrew S. Tanenbaum, and Gary Nutt. The slides are intended for the sole purpose of instruction of operating systems at the University of Rochester. All copyrighted materials belong to their original owner(s).