CS 591: Introduction to Computer Security

Confinement

James Hook

The Confinement Problem

- Lampson, "A Note on the Confinement Problem", CACM, 1973.

  This note explores the problem of confining a program during its execution so that it canot transmit information to any other program except its caller. A set of examples attempts to stake out the boundaries of the problem. Necessary conditions for a solution are stated and informally justified.

Possible Leaks

0. If a service has memory, it can collect data, wait for its owner to call it, then return the data
1. The service may write into a permanent file
2. The service may create a temporary file
3. The service may send a message to a process controlled by its owner [via ipc]
4. More subtly, the information may be encoded in the bill rendered for the service…

Possible Leaks (cont)

5. If the system has interlocks which prevent files from being open for writing and reading at the same time, the service can leak data if it is merely allowed to read files which can be written by the owner.

Leak 5 (cont)

The interlocks allow a file to simulate a shared Boolean variable which one program can set and the other can't

Given a procedure `open (file, error)` which does **`goto`** `error` if the file is already open, the following procedures will perform this simulation:

```
procedure settrue (file);
    begin loop1: open (file, loop1) end;
procedure setfalse (file);
    begin close (file) end;
Boolean procedure value (file);
    begin value : = true;
          open (file, loop2);
          value := false;
          close (file);
      loop2:
    end;
```

5/3/06 15:39

Leak 5 (cont)

Using these procedures and three files called data, sendclock, and receiveclock, a service can send a stream of bits to another concurrently running program. Referencing the files as though they were variables of this rather odd kind, then, we can describe the sequence of events for transmitting a single bit:

```
sender:     data : = bit being sent;
            sendclock : = true
receiver:   wait for sendclock = true;
            received bit : = data;
            receive clock : = true;
sender:     wait for receive clock = true;
            sendclock : = false;
receiver:   wait for sendclock = false;
            receiveclock : = false;
sender:     wait for receiveclock = false;
```

5/3/06 15:39

Leak 6

6. By varying its ratio of computing to input/output or its paging rate, the service can transmit information which a concurrently running process can receive by observing the performance of the system.
   …

One solution

- Just say no!
- Total isolation: A confined program shall make no calls on any other program
- Impractical

Confinement rule

- Transitivity: If a confined program calls another program which is not trusted, the called program must also be confined.

Classification of Channels:

- Storage
- Legitimate (such as the bill)
- Covert
  - I.e. those not intended for information transfer at all, such as the service program's effect on the system load
- In which category does Lampson place 5?

Root Problem:

- Resource sharing enables covert channels
- The more our operating systems and hardware enable efficient resource sharing the greater the risk of covert channels

5/3/06 15:39

Resources

- Lampson, A note on the Confinement Problem, CACM Vol 16, no. 10, October 1973.
  - http://doi.acm.org/10.1145/362375.362389

5/3/06 15:39

Discussion

- Bishop's slides for Chapter 16 (with some minor modifications to one example)

Virtualization

- Virtualization is returning to the mainstream with Intel's Virtualization Technology (aka Vanderpool)
- Discussion following Bishop's slides for Chapter 29
  - Secret decoder ring:
    - PSL = Processor Status Longword (a vax status register)

Applications of Virtualization

- Workload isolation
- Workload consolidation
- Workload migration
- (See Uhlig, et al, Fig 1)

Virtualizing Intel architectures

- As is, Intel architectures do not meet the two requirements:
  - Nonfaulting access to privileged state
    - IA-32 has registers that describe and manipulate the "global descriptor table"
    - These registers can only be set in ring 0
    - They can be queried in any ring without generating a fault
  - This violates rule 2 (all references to sensitive data traps)
- Software products to virtualize Intel hardware had to get around this.
  - Vmware dynamically rewrote code!

Intel solutions

- VT-x, virtualization for IA-32
- VT-i, virtualization for Itanium
- Changed architecture to meet the criteria

5/3/06 15:39

Ring aliasing and ring compression

- Solution is to allow guest to run at intended privilege level by augmenting privilege levels.
- See Figure 2(d).

5/3/06 15:39

Nonfaulting access to privileged state

- Two kinds of changes
  - Make access fault to the VM
  - Allow nonfaulting access, but to state under the control of the VMM

- Intel Virtualization Paper
  - ftp://download.intel.com/technology/computing/vptech/vt-ieee-computer-final.pdf