ECS 289M Lecture 20

May 17, 2006

Multiple Levels of Privilege

- Hardware supports *n* levels of privilege
  - So each VM must appear to do this also
- But only VMM can run at highest level
  - So *n*–1 levels available to each VM
- VMs must virtualize levels of privilege
  - Technique called *ring compression*

Example: VAX/VMM

- VMM must emulate 4 levels of privilege
  - Cannot allow any VM to enter kernel mode, and thereby bypass VMM
  - But VAX/VMS requires all four levels!
- Virtualize executive, kernel privilege levels
  - Conceptually, map both to physical executive level
  - Add VM bit to PSL; if set, current process is on VM
  - VMPSL register records PSL of running VM
  - All sensitive instructions obtain info from VMPSL or trap to VMM, which emulates instruction

Another Approach

- Divide users into different classes
  - Control access to system by limiting access of each class
- Example: IBM VM/370 associates various commands with users
  - Each command associated with *user privilege classes*
    - Class G ("general user") can start VM
    - Class A ("primary system operator") can control system accounting, availability of VMs, etc.
    - Class "Any" can access, relinquish access, to VM

Physical Resources and VMs

- VMM distributes these among VMs as appropriate
- Example: minidisks
  - System to run 10 VMs using one disk
  - Split disk into 10 minidisks
  - VMM handles mapping from (virtual) minidisk address to physical disk address

Example

- VM's OS tries to write to a disk
  - Privileged I/O instruction causes trap to VMM
  - VMM translates address in I/O instruction to address in physical disk
  - VMM checks that physical address in area of disk allocated to the VM making request
    - If not, request fails; error returned to VM
  - VMM services request, returns control to VM

Paging and VM

- Paging on ordinary machines is at highest privilege level
- Paging on VM is at highest virtual level
  - Handled like any other disk I/O
- Two problems:
  - On some machines, some pages available only from highest privilege level, but VM runs at next-to-highest level
  - Performance

First Problem

- VM must change protection level of pages available only from highest privilege level to appropriate level
- Example:
  - On VAX/VMS, kernel mode needed for some pages
  - But VM runs at executive mode, so must ensure only virtual kernel level processes can read those pages
  - In practice, VMS system allows executive mode processes to elevate to kernel mode; no security issue
  - But … executive mode processes on non-VM system cannot read pages, so loss of reliability

Second Problem

- VMM pages: transparent to VMs
- VMs page: VMM handles it as above
  - If lots of VM paging, this may cause significant delay
- Example: IBM VM/370
  - OS/MFT, OS/MVT access disk storage
    - If jobs depend on timings, delays caused by VMM may affect results
  - MVS does that and pages, too
    - Jobs depending on timings could fail under VM/370 that would succeed if run under MVS directly

VMM as Security Kernel

- VMM deals with subjects (the VMs)
  - Knows nothing about the processes within the VM
- VMM applies security checks to subjects
  - By transitivity, these controls apply to processes on VMs
- Thus, satisfies rule of transitive confinement

Example 1: KVM/370

- KVM/370 is security-enhanced version of VM/370 VMM
  - Goal: prevent communications between VMs of different security classes
  - Like VM/370, provides VMs with minidisks, sharing some portions of those disks
  - Unlike VM/370, mediates access to shared areas to limit communication in accordance with security policy

Example 2: VAX/VMM

- Can run either VMS or Ultrix
- 4 privilege levels for VM system
  - VM user, VM supervisor, VM executive, VM kernel (both physical executive)
- VMM runs in physical kernel mode
  - Only it can access certain resources
- VMM subjects: users and VMs

Example 2

- VMM has flat file system for itself
  - Rest of disk partitioned among VMs
  - VMs can use any file system structure
    - Each VM has its own set of file systems
  - Subjects, objects have security, integrity classes
    - Called *access classes*
  - VMM has sophisticated auditing mechanism

Problem

- Physical resources shared
  - System CPU, disks, etc.
- May share logical resources
  - Depends on how system is implemented
- Allows covert channels

Sandboxes

- An environment in which actions are restricted in accordance with security policy
  - Limit execution environment as needed
    - Program not modified
    - Libraries, kernel modified to restrict actions
  - Modify program to check, restrict actions
    - Like dynamic debuggers, profilers

Examples Limiting Environment

- Java virtual machine
  - Security manager limits access of downloaded programs as policy dictates
- Sidewinder firewall
  - Type enforcement limits access
  - Policy fixed in kernel by vendor
- Domain Type Enforcement
  - Enforcement mechanism for DTEL
  - Kernel enforces sandbox defined by system administrator

Modifying Programs

- Add breakpoints or special instructions to source, binary code
  - On trap or execution of special instructions, analyze state of process
- Variant: *software fault isolation*
  - Add instructions checking memory accesses, other security issues
  - Any attempt to violate policy causes trap

Example: Janus

- Implements sandbox in which system calls checked
  - *Framework* does runtime checking
  - *Modules* determine which accesses allowed
- Configuration file
  - Instructs loading of modules
  - Also lists constraints

Configuration File

```
# basic module
basic

# define subprocess environment variables
putenv IFS="\t\n " PATH=/sbin:/bin:/usr/bin TZ=PST8PDT

# deny access to everything except files under /usr
path deny read,write *
path allow read,write /usr/*
# allow subprocess to read files in library directories
# needed for dynamic loading
path allow read /lib/* /usr/lib/* /usr/local/lib/*
# needed so child can execute programs
path allow read,exec /sbin/* /bin/* /usr/bin/*
```


How It Works

- Framework builds list of relevant system calls
  - Then marks each with allowed, disallowed actions
- When monitored system call executed
  - Framework checks arguments, validates that call is allowed for those arguments
    - If not, returns failure
    - Otherwise, give control back to child, so normal system call proceeds

Use

- Reading MIME Mail: fear is user sets mail reader to display attachment using Postscript engine
  - Has mechanism to execute system-level commands
  - Embed a file deletion command in attachment …
- Janus configured to disallow execution of any subcommands by Postscript engine
  - Above attempt fails

Sandboxes, VMs, and TCB

- Sandboxes, VMs part of trusted computing bases
  - Failure: less protection than security officers, users believe
  - “False sense of security”
- Must ensure confinement mechanism correctly implements desired security policy