Lecture 08 – Access Control and Isolation

Michael Bailey
University of Illinois
ECE 422/CS 461 – Spring 2018

Authentication vs Authorization

- Authentication — Who goes there?
  - Restrictions on who (or what) can access system
- **Authorization** — Are you allowed to do that?
  - Restrictions on actions of authenticated users
- Authorization is a form of **access control**
- Authorization enforced by
  - Access Control Lists
  - Capabilities

Access Control

- Access control is a collection of methods and components that supports
  - confidentiality
  - integrity
- Goal: allow only authorized subjects to access permitted objects
- E.g., Least privilege philosophy

  A subject is granted permissions needed to accomplish required tasks and nothing more

Access Control Designs

- Access control designs define rules for users accessing files or devices
- Three common access control designs
  - Mandatory access control
  - Discretionary access control
  - Role-based access control

Mandatory Access Control

- It is a restrictive scheme that does not allow users to define permissions on files, regardless of ownership.
- Instead, security decisions are made by a central policy administrator.
- A common implementation is rule-based access control
  - Subject demonstrates need-to-know in addition to proper security clearance
  - Need-to-know indicates that a subject requires access to object to complete a particular task
- **Security-Enhanced Linux (SELinux)** incorporates mandatory access control.

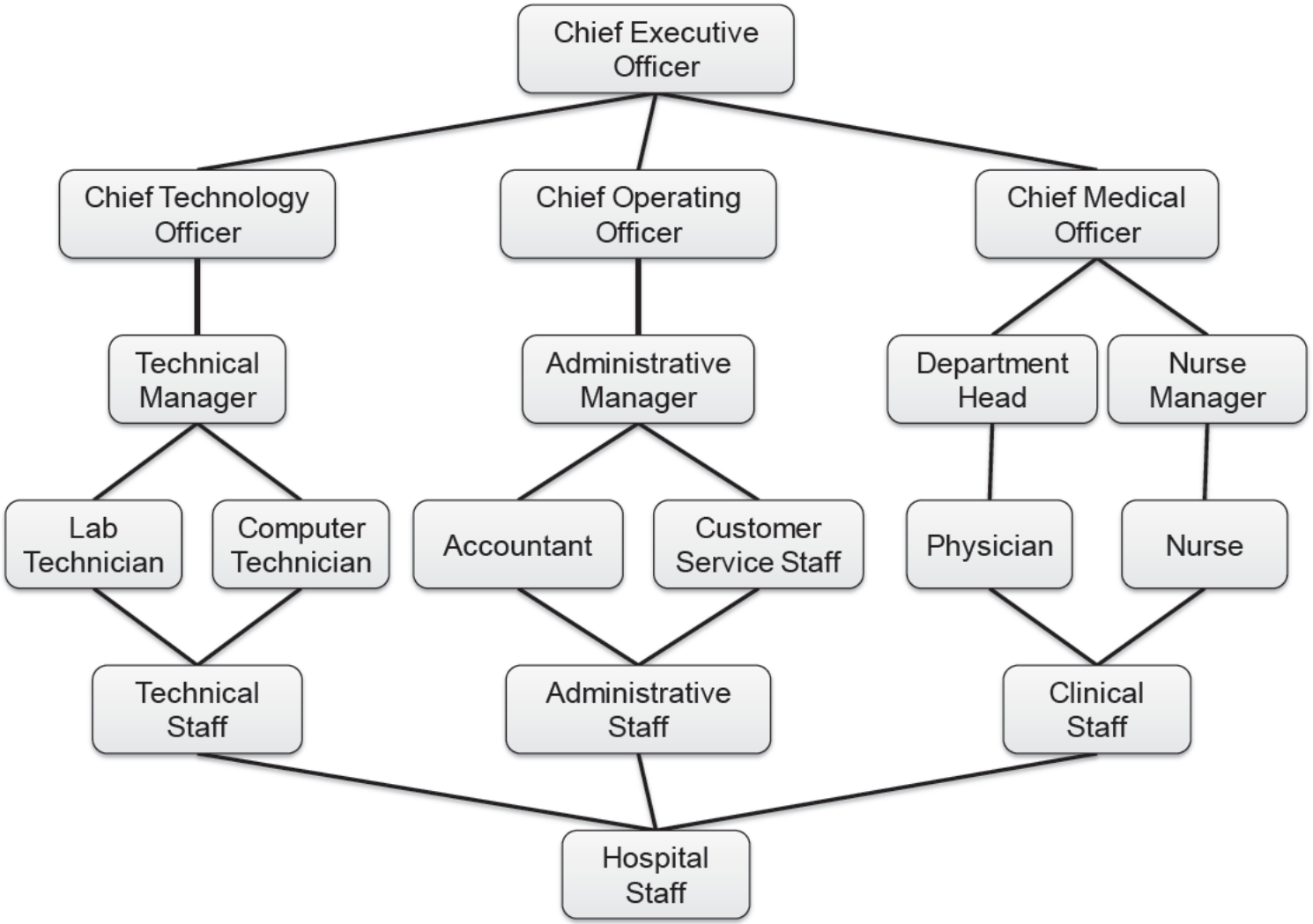
Discretionary Access Control

- Discretionary access control, or **DAC,** refers to a scheme where users are given the ability to determine the permissions governing access to their own files.
  - DAC typically features the concept of both users and groups
  - In addition, DAC schemes allow users to grant privileges on resources to other users on the same system.
- Most common design in commercial operating systems
  - Generally less secure than mandatory control
  - Generally easier to implement and more flexible

Role-Based Access Control

- The **role-based access control (RBAC)** model can be viewed as an evolution of the notion of group-based permissions in file systems.
- An RBAC system is defined with respect to an organization, such as company, a set of resources, such as documents, print services, and network services, and a set of users, such as employees, suppliers, and customers
- Uses a subject's role or task to grant or deny object access
  - In the role-

Visualizing Role Hierarchy

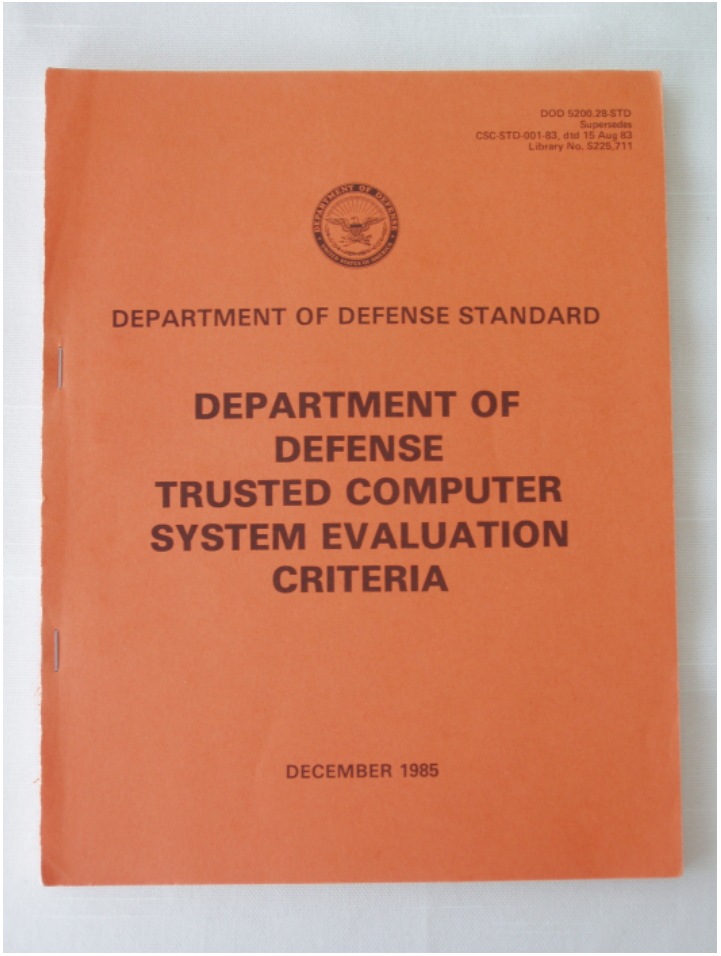


E.g., Best Practices of Information Classification

- Military classifications of access rights for documents based on concepts
  - Unclassified
  - Confidential
  - Secret
  - Top secret

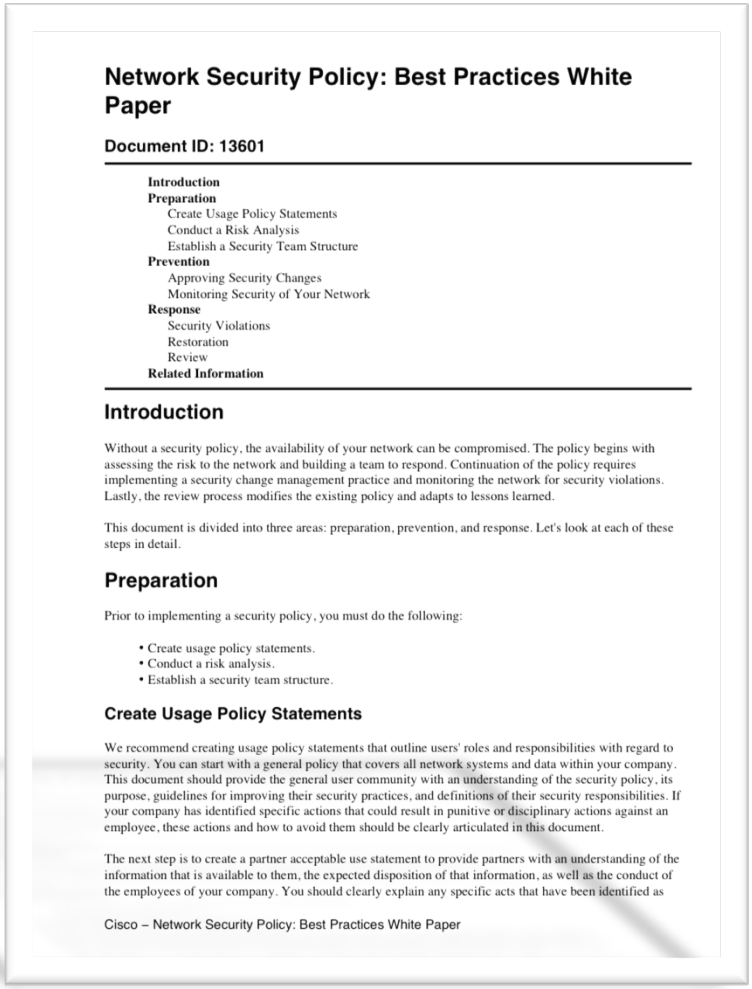
E.g., The Orange Book

- Trusted Computer System Evaluation Criteria (TCSEC)
  - Division D: “minimal protection”
  - Division C: “Discretionary protection”
  - Division B: “Mandatory protection”
  - Division A: “Verified protection”



E.g., Cisco Best Practices

- Preparation
  - Create Usage Policy Statement
  - Conduct a Risk Analysis
  - Establish a Security Team Structure
- Prevention
  - Approving Security Changes
  - Monitoring Security of Your Network
- Response
  - Security Violations
  - Restoration
  - Review



Network Security Policy: Best Practices White Paper

Document ID: 13601

Introduction

Without a security policy, the availability of your network can be compromised. The policy begins with assessing the risk to the network and building a team to respond. Continuation of the policy requires implementing a security change management practice and monitoring the network for security violations. Lastly, the review process modifies the existing policy and adapts to lessons learned.

This document is divided into three areas: preparation, prevention, and response. Let's look at each of these steps in detail.

Preparation

Prior to implementing a security policy, you must do the following:

- Create usage policy statements.
- Conduct a risk analysis.
- Establish a security team structure.

Create Usage Policy Statements

We recommend creating usage policy statements that outline users' roles and responsibilities with regard to security. You can start with a general policy that covers all network systems and data within your company. This document should provide the general user community with an understanding of the security policy, its purpose, guidelines for improving their security practices, and definitions of their security responsibilities. If your company has identified specific actions that could result in punitive or disciplinary actions against an employee, these actions and how to avoid them should be clearly articulated in this document.

The next step is to create a partner acceptable use statement to provide partners with an understanding of the information that is available to them, the expected disposition of that information, as well as the conduct of the employees of your company. You should clearly explain any specific acts that have been identified as

Cisco – Network Security Policy: Best Practices White Paper

Example Threat and Policy

Lost Devices

Data Leakage by Lost Devices

Council confidential data loss causes ICO concern

By Arwyn Jones
BBC News Wales

Welsh councils are not doing enough to protect people's confidential data from falling into the wrong hands, according to the UK information watchdog.

containing
councils

Department	Number of records lost	Narrative
Department for Work and Pensions	n/a	USB memory stick, apparently encrypted and containing passwords for an old version of the Government Gateway, a website giving access to millions of records of personal data.
Ministry of Defence	1,700,000	Hard drive being held by contractor EDS is found to be missing.
Service Personnel and Veterans Agency	50,500	Three USB portable hard drives with details of staff are allegedly stolen from a high security facility at RAF Innsworth. The Agency holds records on 900,000 current and former personnel. Stolen records included sensitive information about the private lives of senior staff.
Insolvency Service	400	Names, addresses and bank details of up to 400 directors of 122 firms were lost when four laptops were stolen from a Manchester premises.
Tees, Esk and Wear Valleys NHS Trust	200	Memory stick with details of patients found in a public park.
Home Office	84,000	PA Consulting lost an unencrypted memory stick containing details high risk, prolific and other offenders.
Colchester Hospital University NHS Foundation Trust	21,000	A manager's unencrypted laptop holding patient addresses and treatment details is stolen from his car whilst on holiday in Edinburgh
Department for Work and Pensions	45,000	West Yorkshire benefit claimants' in data lost.
Department for Work and Pensions	000s	CDs with personal data found at the home of a former contractor.
City and Hackney Teaching Primary Care Trust	160,000	"Heavily encrypted" disks containing details of children are lost by couriers. The loss prompted the agency to implement hard drive and USB memory stick encryption systems across all PCs.
Foreign and Commonwealth Office	50,000	Details of visa applicants were made available on an FCO website.
HM Revenue and Customs	25,000,000	Two CDs containing details of the families of child benefits claimants went missing in the post. HMRC's handling of data was described as "woefully inadequate" and staff were described as "muddling through" in a June 2008 Independent Police Complaints Commission report.
Ministry of Justice	5,000	Hard disk with details of HM Prison Service staff is lost on the premises of EDS.
Driving Standards Agency	3,000,000	Hard disk with details of candidates for the driving theory test was lost in a premises in Iowa by subcontractors.
Foreign and Commonwealth Office	50	Details of individuals made public after "unauthorised disclosure by a contractor"

Glue it?


1. Remove cap
2. Insert glue nozzle into USB port and apply liberally
3. Kick back and relax, your endpoint security problems are over!


HOT GLUE
ALL THE USB PORTS
memegenerator.net

Fine it?

£120,000 fine for lost USB stick

The ICO has fined Greater Manchester Police £120,000 after a memory stick containing sensitive personal data was stolen from an officer's home.

According to the ICO the device had no encryption or password protection, and contained details of more than a thousand people with links to serious crime investigations.

The ICO found that some officers across the force regularly used unencrypted memory sticks, which may also have been used to copy data from police computers to access away from the office. Despite a similar security breach in September 2010, the force had not put restrictions on downloading information, and staff were not sufficiently trained in data protection.

David Smith, ICO Director of Data Protection, said: "This is a substantial monetary penalty, reflecting the significant failings the force demonstrated. We hope it will discourage others from making the same data protection mistakes."

"It is easy to protect against such risks" says Jon Stanton from PEM IT Services. "The organisation could have installed security software such as DriveLock to control what USB devices can access their PCs. DriveLock would also have enabled them to encrypted their memory sticks, and given them an audit trail of what files had been transferred. This, combined with staff training, would have minimised the risks of any data loss."

Lost Laptops

- Lost and stolen laptops are a common occurrence
  - Estimated occurrences in US airports every week: 12,000
- Average cost of a lost laptop for a corporation is $50K
  - Costs include data breach, intellectual property loss, forensics, lost productivity, legal and regulatory expenses
  - Data breach much more serious than hardware loss
- Data breach cost estimated at $200 per customer record
  - Direct costs include discovery, notification and response
  - Indirect costs include customer turnover (higher loss and lower acquisition)
- Data can also be copied while laptop is unattended

Ponemon Institute. Research Studies & White Papers: Security

From Device to Data – Encryption

- In a perfect world, we would not store sensitive data on portable devices
  - All sensitive data should be held on secure servers.
  - Unfortunately, this approach is not always practical.
- Keep the benefit of using portable devices
- Reduce the risk of data leakage by encryption

Encryption of File Systems

- Disk encryption
  - Block-level encryption
  - Encryption of physical or logical drive
  - BitLocker in Windows Vista and 7
  - TrueCrypt open source software
- File system encryption
  - File-level encryption
  - Encrypting File System (EFS) in Windows

Example Threat and Policy

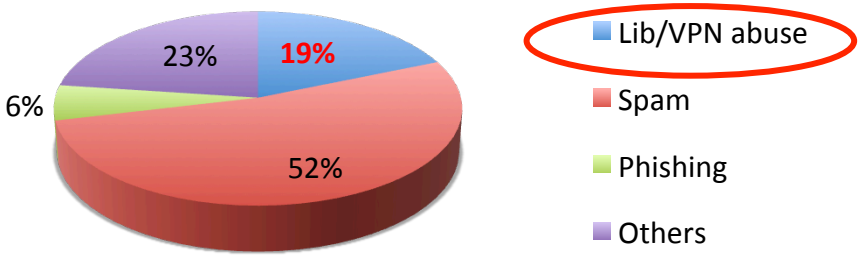
Passwords

Warning to the Users

- Do not use shared credentials
- Strength your password
  - 2,295: “123456” or a sequential list of number
  - 780: “password”
  - 437: “welcome”

Compromised UofM Accounts

- **613 incidents** related to unauthorized use of university accounts during 2010 and first 6 months of 2011 at UofM



Compromised UofM Accounts

- **What did they do with the compromised accounts?**

  Netflow data analysis

  - ✓ Library website repeatedly visited
  - ✓ 8.2% of HTTP flows visited consist of 10 websites blocked in China
  - ✓ Login to accounts at 7 universities

- **Market place for the compromised university accounts**

New motivation of attackers who steal university credentials:

Free and unfettered access to information

Weak Password Scanning

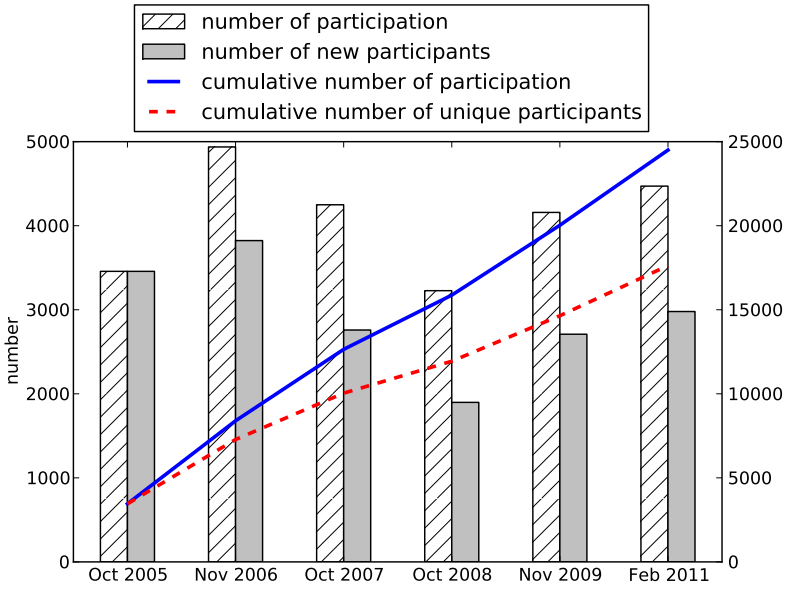
- UofM scan weak password accounts twice a year using a commercial password cracker
- 2,284 weak password are detected in June, 2011
- Whether accounts with weak passwords are a problem?

	# of total	# of compromised	Pr (compromise)
Weak Password	2,284	12	0.525%
Total Population	550,000	380	0.069%

Test statistics of deviance of 28.09 and a p-value of 1.16^{-16}

Educating the Users

- From ‘what tools to use’ to ‘who uses the tool’



Yearly Security Quiz at UofM

	# of total	# of compromised	Pr (compromise)
Passed Quiz	9.227	9	0.1%
NOT passed	41,924	105	0.25%

Test statistics of deviance of 13.52 and a p-value of 2.36^{-4}

Example of Implementing Policy

Filesystem Access Control

Access Control Entries and Lists

- An Access Control List (ACL) for a resource (e.g., a file or folder) is a sorted list of zero or more Access Control Entries (ACEs)
- An ACE refers specifies that a certain set of accesses (e.g., read, execute and write) to the resources is allowed or denied for a user or group
- Examples of ACEs for folder “Bob’s CS167 Grades”
  - Bob; Read; Allow
  - TAs; Read; Allow
  - TWD; Read, Write; Allow
  - Bob; Write; Deny
  - TAs; Write; Allow

Linux File System

- Tree of directories (folders)
- Each directory has links to zero or more files or directories
- Hard link
  - From a directory to a file
  - The same file can have hard links from multiple directories, each with its own filename, but all sharing owner, group, and permissions
  - File deleted when no more hard links to it
- Symbolic link (symlink)
  - From a directory to a target file or directory
  - Stores path to target, which is traversed for each access
  - The same file or directory can have multiple symlinks to it
  - Removal of symlink does not affect target
  - Removal of target invalidates (but not removes) symlinks to it
  - Analogue of Windows shortcut or Mac OS alias

Unix Permissions

- Standard for all UNIXes
- Every file is owned by a user and has an associated group
- Permissions often displayed in compact 10-character notation
- To see permissions, use `ls -l`

```
jk@sphere:~/test$ ls -l
total 0
-rw-r-----  1 jk ugrad 0 2005-10-13 07:18 file1
-rwxrwxrwx  1 jk ugrad 0 2005-10-13 07:18 file2
```

Permissions Examples (Regular Files)

-rw-r--r--	read/write for owner, read-only for everyone else
-rw-r-----	read/write for owner, read-only for group, forbidden to others
-rwx------	read/write/execute for owner, forbidden to everyone else
-r--r--r--	read-only to everyone, including owner
-rwxrwxrwx	read/write/execute to everyone

Permissions for Directories

- Permissions bits interpreted differently for directories
- *Read* bit allows listing names of files in directory, but not their properties like size and permissions
- *Write* bit allows creating and deleting files within the directory
- *Execute* bit allows entering the directory and getting properties of files in the directory
- Lines for directories in `ls -l` output begin with d, as below:

```
jk@sphere:~/test$ ls -l
Total 4
drwxr-xr-x  2 jk ugrad 4096 2005-10-13 07:37 dir1
-rw-r--r--  1 jk ugrad    0 2005-10-13 07:18 file1
```

Permissions Examples (Directories)

drwxr-xr-x	all can enter and list the directory, only owner can add/delete files
drwxrwx---	full access to owner and group, forbidden to others
drwx--x---	full access to owner, group can access known filenames in directory, forbidden to others
-rwxrwxrwx	full access to everyone

Special Permission Bits

- Three other permission bits exist
  - Set-user-ID (“suid” or “setuid”) bit
  - Set-group-ID (“sgid” or “setgid”) bit
  - Sticky bit

Set-user-ID

- Set-user-ID ("suid" or "setuid") bit
  - On executable files, causes the program to run as file owner regardless of who runs it
  - Ignored for everything else
  - In 10-character display, replaces the 4th character (x or -) with s (or S if not also executable)

    -rwsr-xr-x: setuid, executable by all

    -rwxr-xr-x: executable by all, but not setuid

    -rwSr--r--: setuid, but not executable - not useful

Root

- “root” account is a super-user account, like Administrator on Windows
- Multiple roots possible
- File permissions do not restrict root
- This is *dangerous*, but necessary, and OK with good practices

Becoming Root

- su
  - Changes home directory, PATH, and shell to that of root, but doesn't touch most of environment and doesn't run login scripts
- su -
  - Logs in as root just as if root had done so normally
- sudo <command>
  - Run just one command as root
- su [-] <user>
  - Become another non-root user
  - Root does not require to enter password

Changing Permissions

- Permissions are changed with chmod or through a GUI like Konqueror
- Only the file owner or root can change permissions
- If a user owns a file, the user can use chgrp to set its group to any group of which the user is a member
- root can change file ownership with chown (and can optionally change group in the same command)
- chown, chmod, and chgrp can take the -R option to recur through subdirectories

Examples of Changing Permissions

chown -R root dir1	Changes ownership of dir1 and everything within it to root
chmod g+w,o-rwx file1 file2	Adds group write permission to file1 and file2, denying all access to others
chmod -R g=rwX dir1	Adds group read/write permission to dir1 and everything within it, and group execute permission on files or directories where someone has execute permission
chgrp testgrp file1	Sets file1's group to testgrp, if the user is a member of that group
chmod u+s file1	Sets the setuid bit on file1. (Doesn't change execute bit.)

The confinement principle

Running untrusted code

We often need to run buggy/unstrusted code:

- programs from untrusted Internet sites:
  - apps, extensions, plug-ins, codecs for media player
- exposed applications: pdf viewers, outlook
- legacy daemons: sendmail, bind
- honeypots

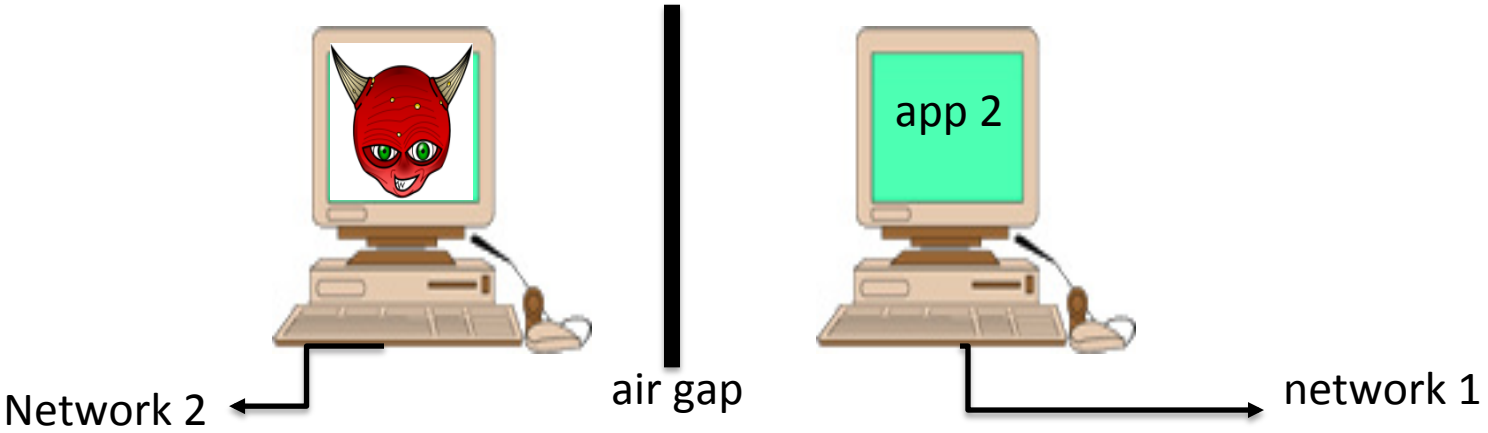
Goal: if application "misbehaves" ⇒ kill it

Approach: confinement

Confinement: ensure misbehaving app cannot harm rest of system

Can be implemented at many levels:

– **Hardware**: run application on isolated hw (air gap)



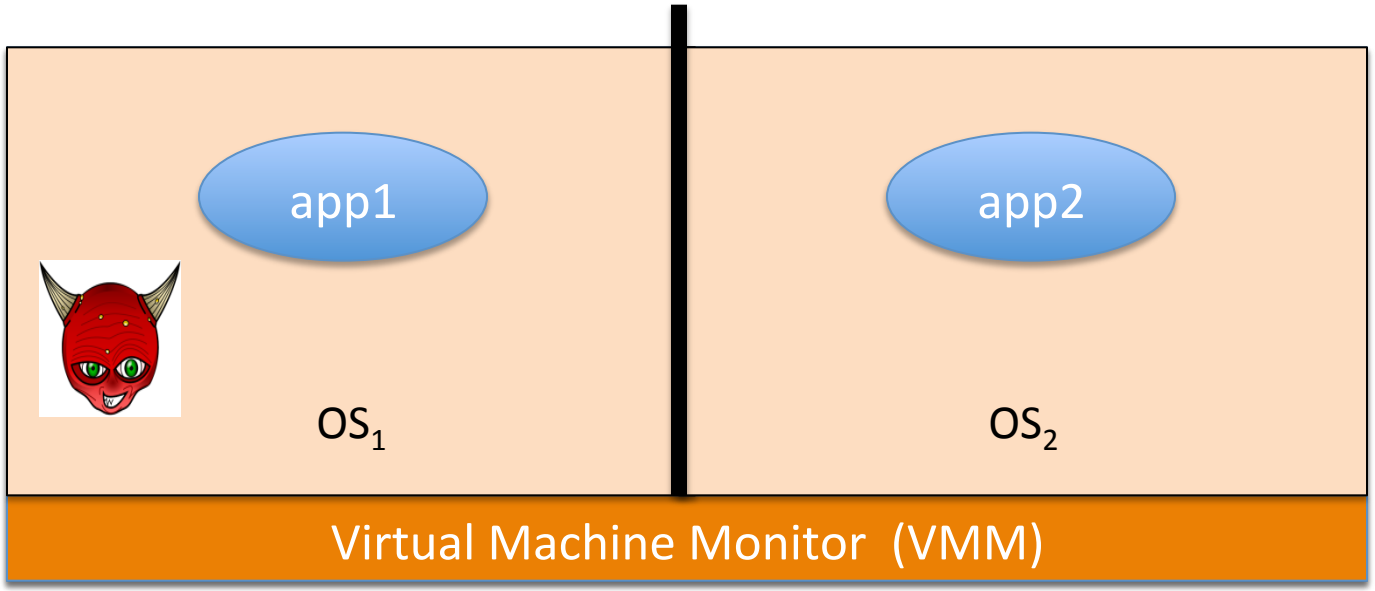
⇒ difficult to manage

Approach: confinement

Confinement: ensure misbehaving app cannot harm rest of system

Can be implemented at many levels:

- **Virtual machines**: isolate OS's on a single machine



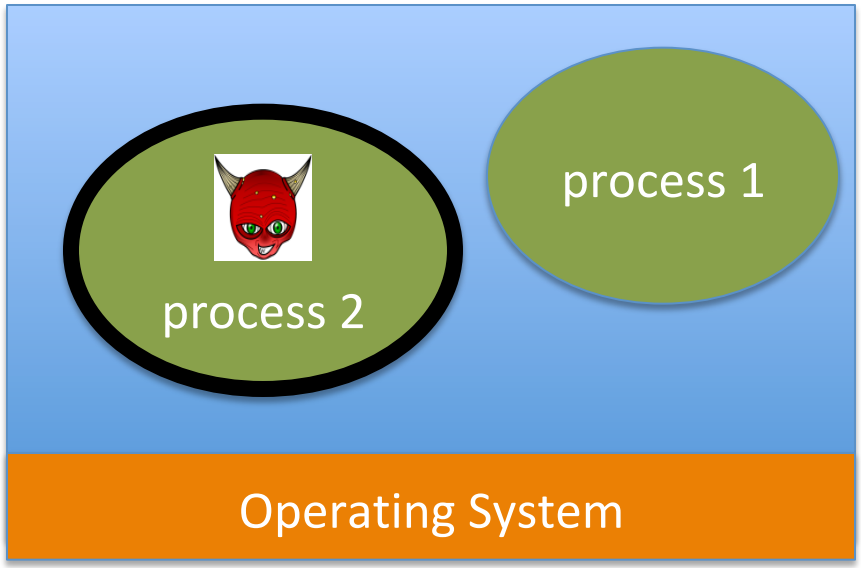
Approach: confinement

Confinement: ensure misbehaving app cannot harm rest of system

Can be implemented at many levels:

- **Process:** System Call Interposition

  Isolate a process in a single operating system



Implementing confinement

Key component: **reference monitor**

- **Mediates requests** from applications
  - Implements protection policy
  - Enforces isolation and confinement
- Must **always** be invoked:
  - Every application request must be mediated
- **Tamperproof**:
  - Reference monitor cannot be killed
  - … or if killed, then monitored process is killed too
- **Small** enough to be analyzed and validated

A old example: chroot

Often used for “guest” accounts on ftp sites

To use do: (must be root)

chroot /tmp/guest	root dir “/” is now “/tmp/guest”
su guest	EUID set to “guest”

Now “/tmp/guest” is added to file system accesses for applications in jail

open(“/etc/passwd”, “r”) ⇒

open(“/tmp/guest/etc/passwd”, “r”)

⇒ application cannot access files outside of jail

Jailkit

Problem: all utility progs (ls, ps, vi) must live inside jail

- **jailkit** project: auto builds files, libs, and dirs needed in jail env
  - **jk_init**: creates jail environment
  - **jk_check:** checks jail env for security problems
    - checks for any modified programs,
    - checks for world writable directories, etc.
  - **jk_lsh**: restricted shell to be used inside jail

- **note:** simple chroot jail does not limit network access

Escaping from jails

Early escapes: relative paths

open("../../etc/passwd", "r") ⇒

open("/tmp/guest/../../etc/passwd", "r")

---

chroot should only be executable by root.

- otherwise jailed app can do:
  - create dummy file "/aaa/etc/passwd"
  - run chroot "/aaa"
  - run su root to become root

(bug in Ultrix 4.0)

Problems with chroot and jail

Coarse policies:

- All or nothing access to parts of file system
- Inappropriate for apps like a web browser
  - Needs read access to files outside jail (e.g. for sending attachments in Gmail)

Does not prevent malicious apps from:

- Accessing network and messing with other machines
- Trying to crash host OS

System Call Interposition

System call interposition

Observation: to damage host system (e.g. persistent changes) app must make system calls:

- To delete/overwrite files: unlink, open, write
- To do network attacks: socket, bind, connect, send

Idea: monitor app's system calls and block unauthorized calls

Implementation options:

- Completely kernel space (e.g. GSWTK)
- Completely user space (e.g. program shepherding)
- Hybrid (e.g. Systrace)

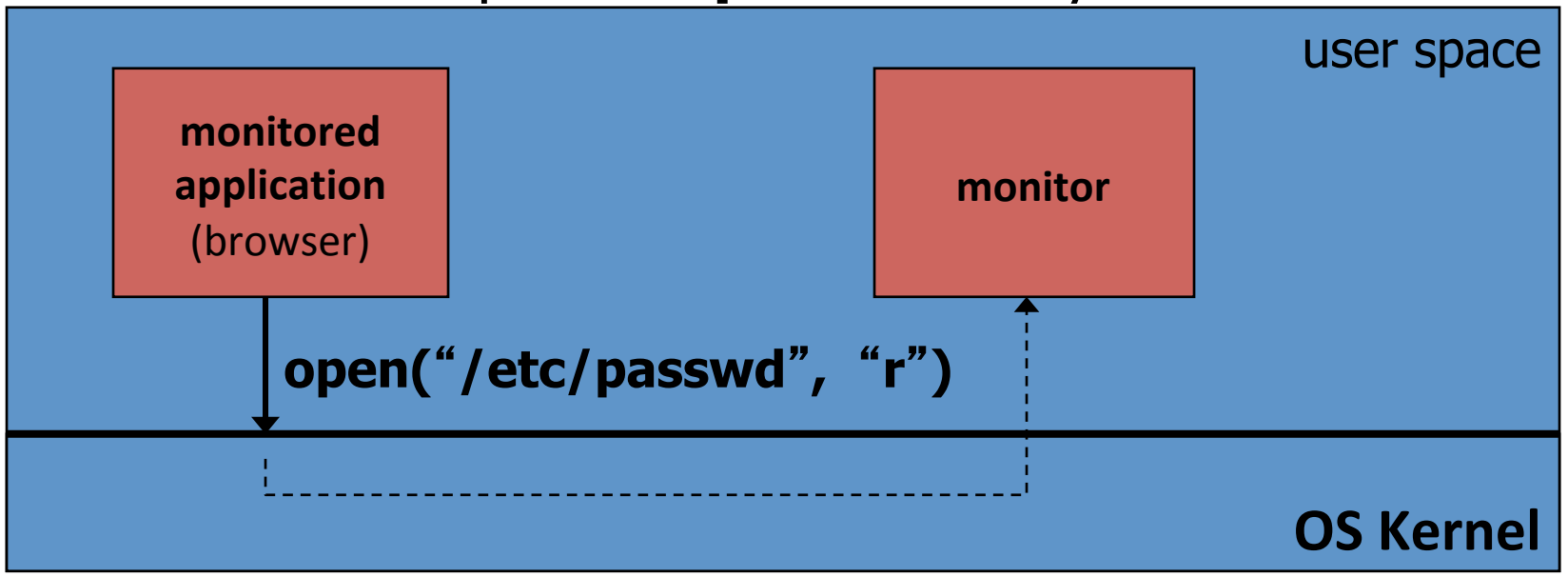
Initial implementation (Janus)

[GWTB'96]

Linux **ptrace**: process tracing

process calls: **ptrace (… , pid_t pid , …)**

and wakes up when **pid** makes sys call.



Monitor kills application if request is disallowed

Complications

```
cd("/tmp")
open("passwd", "r")

cd("/etc")
open("passwd", "r")
```

- If app forks, monitor must also fork
  - forked monitor monitors forked app
- If monitor crashes, app must be killed
- Monitor must maintain all OS state associated with app
  - current-working-dir (**CWD**), **UID, EUID, GID**
  - When app does "cd path" monitor must update its CWD
    - otherwise: relative path requests interpreted incorrectly

Problems with ptrace

Ptrace is not well suited for this application:

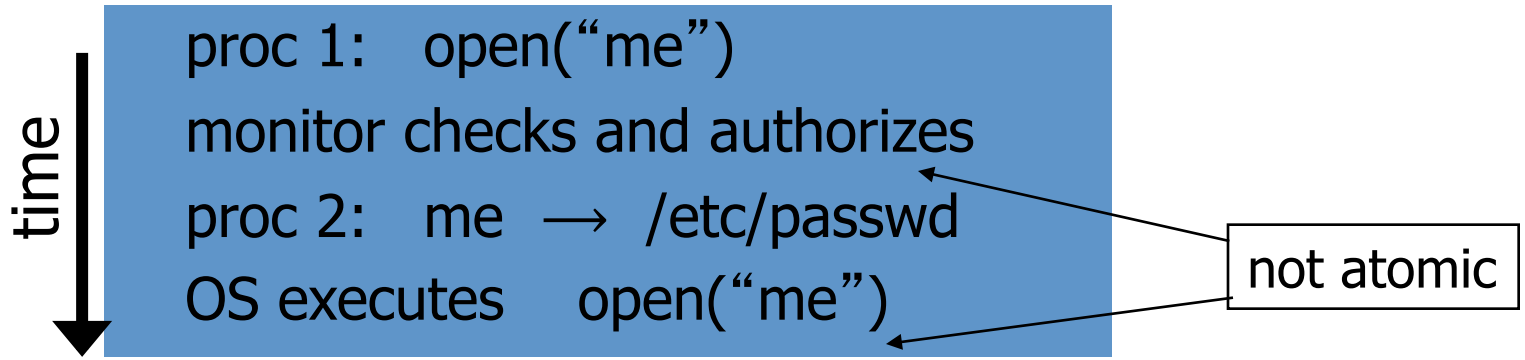
- Trace all system calls or none

  inefficient: no need to trace "close" system call

- Monitor cannot abort sys-call without killing app

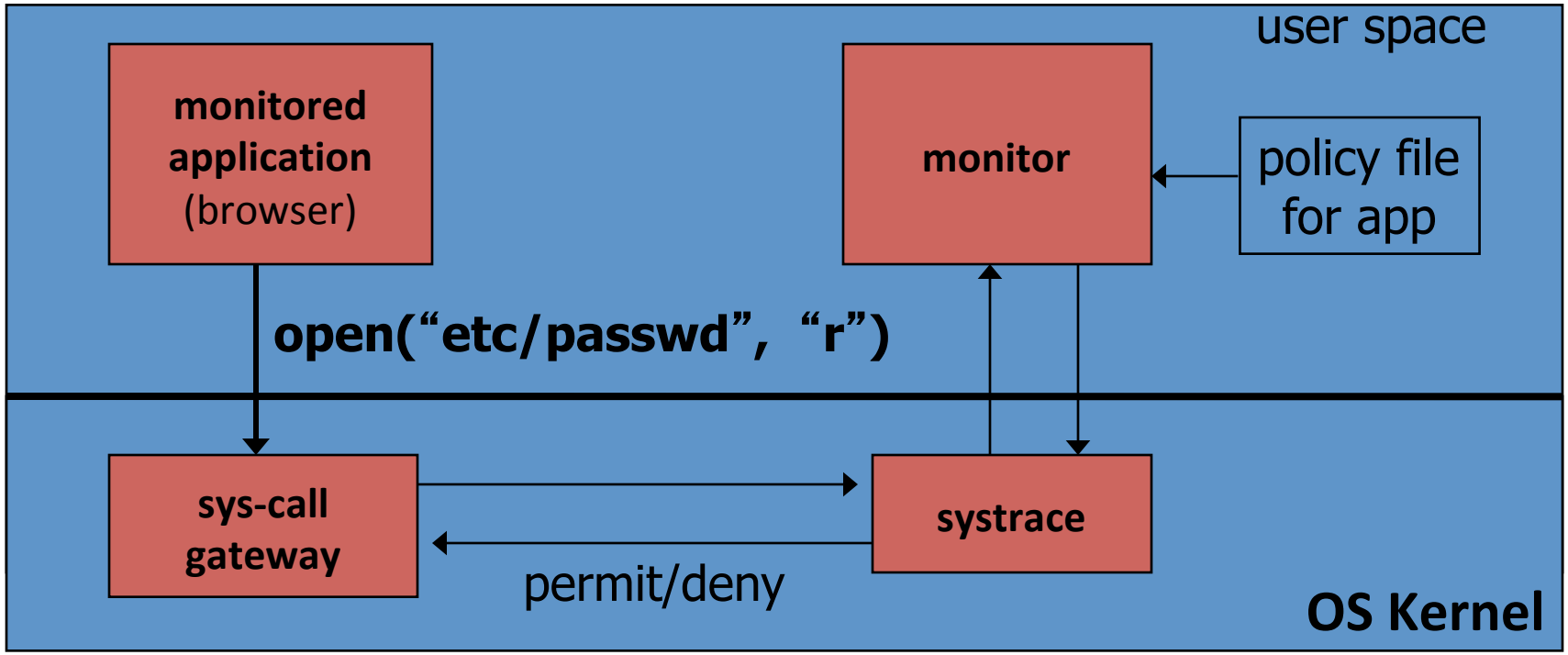
Security problems: **race conditions**

- Example: symlink: me ⟶ mydata.dat



Classic **TOCTOU bug**: time-of-check / time-of-use

Alternate design: systrace [P'02]



- systrace only forwards monitored sys-calls to monitor (efficiency)
- systrace resolves sym-links and replaces sys-call path arguments by full path to target
- When app calls execve, monitor loads new policy file

Policy

Sample policy file:

```
path allow  /tmp/*
path deny  /etc/passwd
network deny all
```

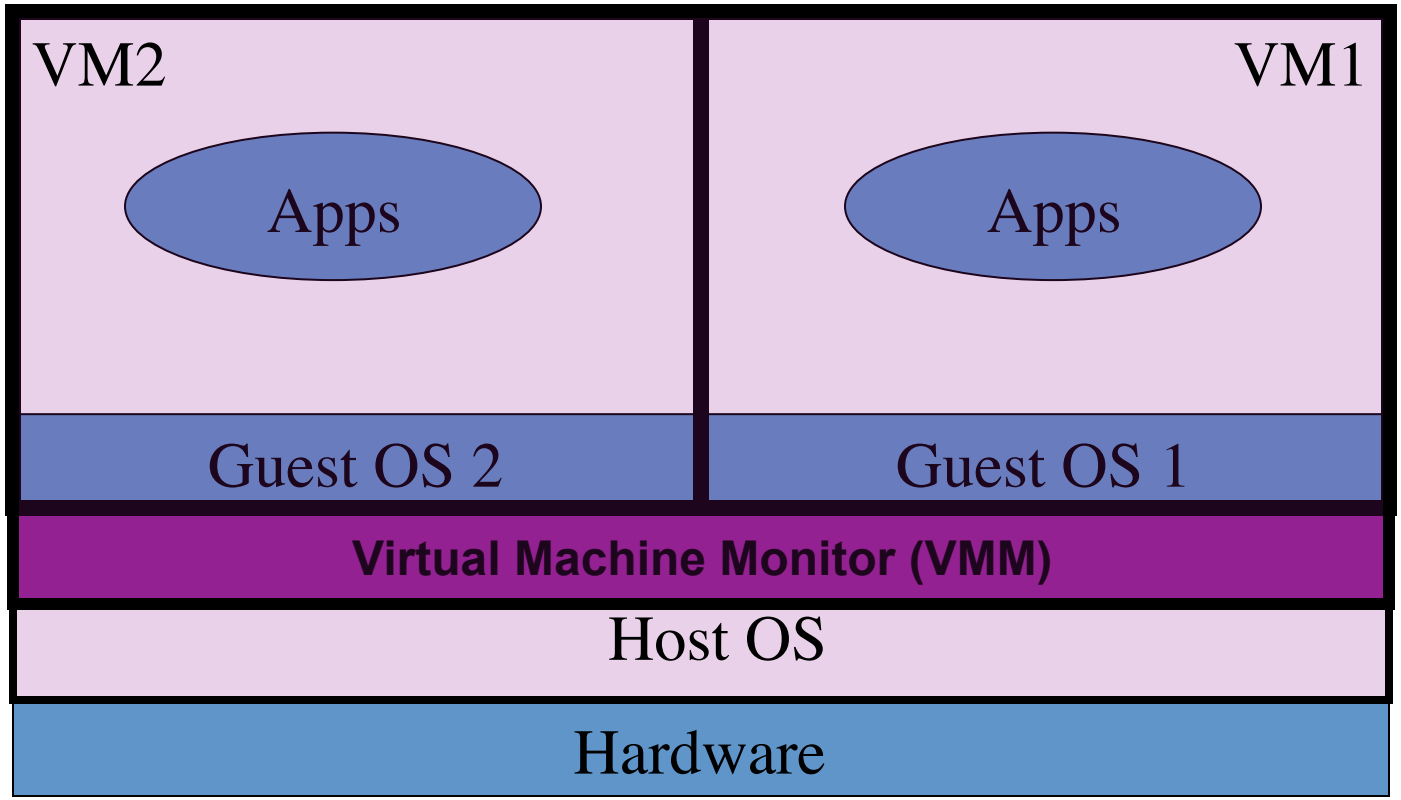
Manually specifying policy for an app can be difficult:

- Systrace can auto-generate policy by learning how app behaves on “good” inputs
- If policy does not cover a specific sys-call, ask user
  - … but user has no way to decide

Difficulty with choosing policy for specific apps (e.g. browser) is the main reason this approach is not widely used

Isolation via Virtual Machines

Virtual Machines



Example: **NSA NetTop**

single HW platform used for both classified and unclassified data

Why so popular now?

VMs in the 1960's:

- Few computers, lots of users
- VMs allow many users to shares a single computer

VMs 1970's – 2000: non-existent

VMs since 2000:

- Too many computers, too few users
  - Print server, Mail server, Web server, File server, Database , …
- Wasteful to run each service on different hardware
- More generally: VMs heavily used in cloud computing

VMM security assumption

VMM Security assumption:

- Malware can infect guest OS and guest apps
- But malware cannot escape from the infected VM
  - Cannot infect host OS
  - Cannot infect other VMs on the same hardware

Requires that VMM protect itself and is not buggy

- VMM is much simpler than full OS

  … but device drivers run in Host OS

Intrusion Detection / Anti-virus

Runs as part of OS kernel and user space process

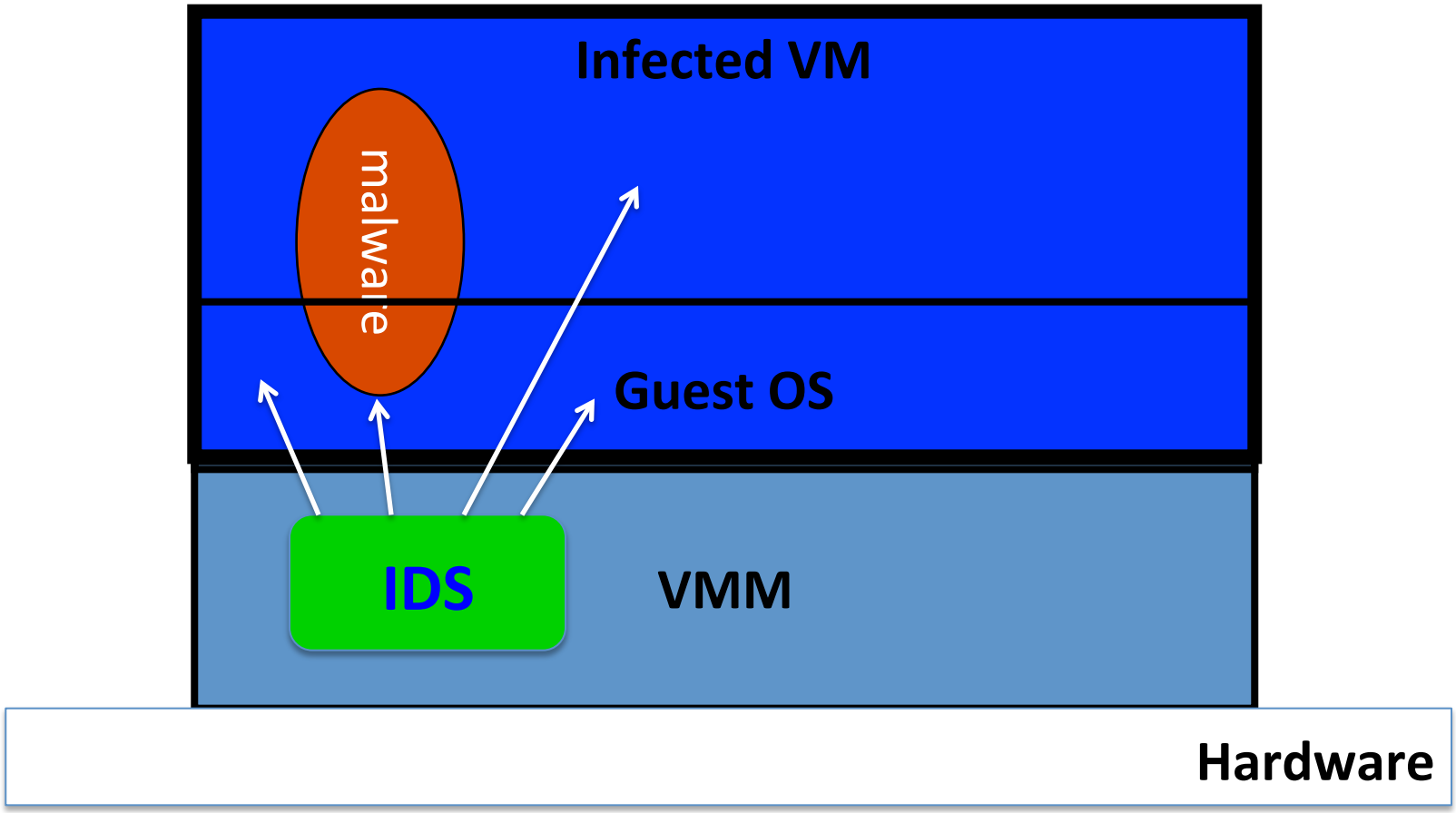
- Kernel root kit can shutdown protection system
- Common practice for modern malware

Standard solution: **run IDS system in the network**

- Problem: insufficient visibility into user's machine

Better: **run IDS as part of VMM (protected from malware)**

- VMM can monitor virtual hardware for anomalies
- VMI: Virtual Machine Introspection
  - Allows VMM to check Guest OS internals


Infected VM
malware
Guest OS
IDS
VMM
Hardware

Sample checks

Stealth root-kit malware:

- Creates processes that are invisible to "ps"
- Opens sockets that are invisible to "netstat"

1. **Lie detector check**
   - Goal: detect stealth malware that hides processes and network activity
   - Method:
     - VMM lists processes running in GuestOS
     - VMM requests GuestOS to list processes (e.g. ps)
     - If mismatch: kill VM

Sample checks

2. **Application code integrity detector**
   - VMM computes hash of user app code running in VM
   - Compare to whitelist of hashes
     - Kills VM if unknown program appears

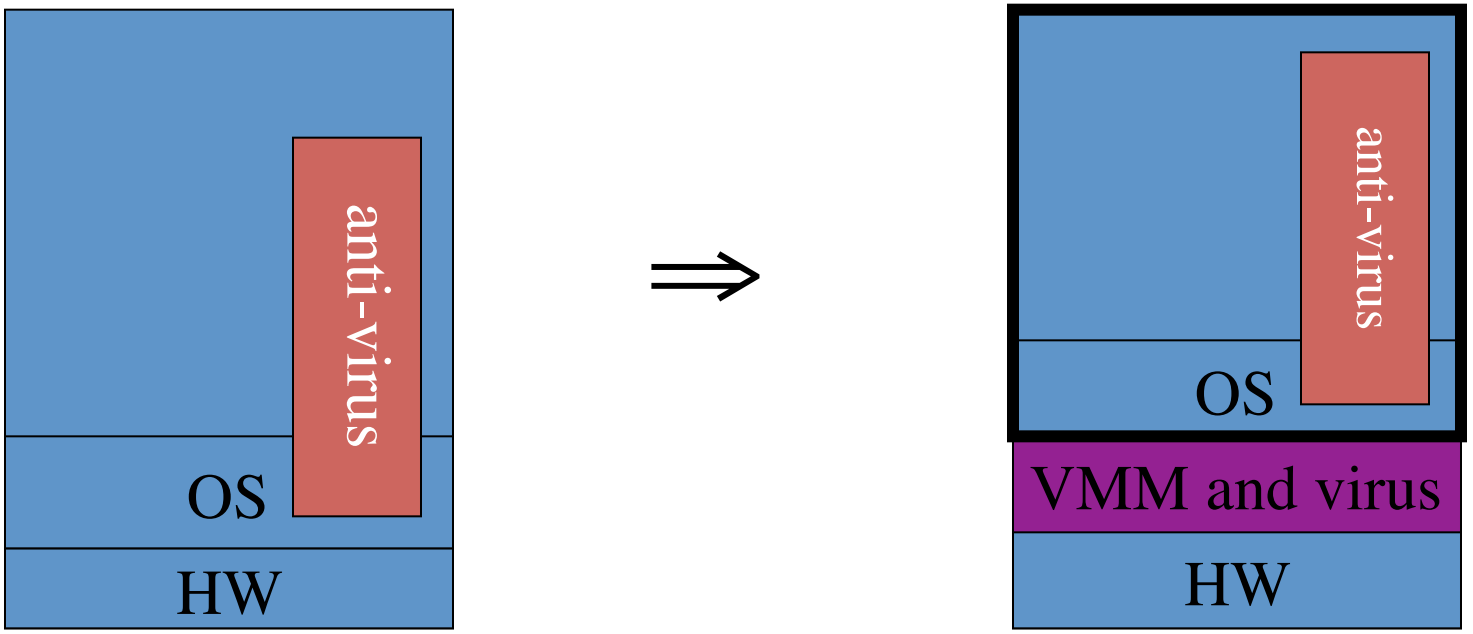
3. **Ensure GuestOS kernel integrity**
   - example: detect changes to sys_call_table

4. **Virus signature detector**
   - Run virus signature detector on GuestOS memory

Problem: Subvirt [King et al. 2006]

Virus idea:

- Once on victim machine, install a malicious VMM
- Virus hides in VMM
- Invisible to virus detector running inside VM



Problem: covert channels

- **Covert channel**: unintended communication channel between isolated components
  - Can be used to leak classified data from secure component to public component