May 15: Information Flow and Confinement

- Information flow for integrity policies
- Examples of information flow controls
  - Android phone
  - Firewalls
- Confinement
- Virtual machines

Integrity Mechanisms

- Biba: mathematical dual of Bell-LaPadula
- Same idea for all constraints, but the opposite
- In general: reverse direction of $\leq$ and reoplace *lub* with *glb*

Assignment

```
x := y + z;
```

Information flows from y, z to x, so for integrity this requires $\underline{x} \le glb(\underline{y}, \underline{z})$

More generally:

$$y := f(x_1, \ldots, x_n)$$

the relation $\underline{y} \le glb\ (\underline{x}_1, \ldots, \mathrm{x}_n)$ must hold

Conditional Statement

```
if x + y < z then a := b else d := b * c - x;
```

- The statement executed reveals information about x, y, z, so $lub(\underline{a}, \underline{d}) \leq glb(\underline{x}, \underline{y}, \underline{z})$

More generally:

`if` $f(x_1, ..., x_n)$ `then` S_1 `else` S_2`; end`

- S_1, S_2 must be certified with respect to integrity
- $lub(\underline{y} \mid y$ target of assignment in $S_1, S_2) \leq$

  $glb(\underline{x}_1, \ldots, \underline{x}_n)$

Example: Android Cellphones

- Usually apps ask for (and get) all permissions
- Ad libraries part of app, so have same permissions
- So app (and libraries) can access information on, about phone
  - Like address book

Information Flow!

- Here, information flowing illicitly out of phone
- So, how do we analyze this?
- Biba, with 2 integrity levels
  - Untainted (U)
  - Tainted (T)
  - T < U (ie, information can flow from untainted to tainted but not the other way)

Example Tool

- TaintDroid: dynamic flow analysis tool
  - Android native libraries are U
  - Those that communicate info externally are *taint sinks*
  - Objects are U or T, as these propagate throughout the system
  - A T object involving a taint sink: data going out of taint sink recorded

During App Operation

- Info flow rules (for integrity) modify tags as rules dictate
  - Android native libraries: external variables referenced, return values tagged based on knowledge of what the code does
- IPC: values in messages grouped by level
- Files: taint tag updated as file written; tag of file tied to variables as file is readf
- Sensors: tagged depending on data

Effectiveness

- Out of 30 popular apps that made 105 network connections using data marked T
  - 2 sent cellphone ID info (like phone number) to server
  - 9 send device identifiers (2 didn’t notify the user they were doing this)
  - 15 sent location info to third parties (none notified the user they were doing this)

Firewalls

- Host that mediates access to a network
  - Blocks or allows access based on security policy
  - If rules applied at the packet level, *packet filtering firewall*
  - If rules applied at the application level, *proxy* or *application level firewall*
  - If it keeps track of state of each connection, it's a *stateful firewall*

Examples

- Firewall checks all incoming email for malware, and discards letters with that
- Java applet coming from an untrusted source
  - On each HTTP connection, firewall analyzes connection to see if applet coming over
  - If so, analyze the applet to see if it is safe; discard the applet; or disable it (change “<applet>” to something else

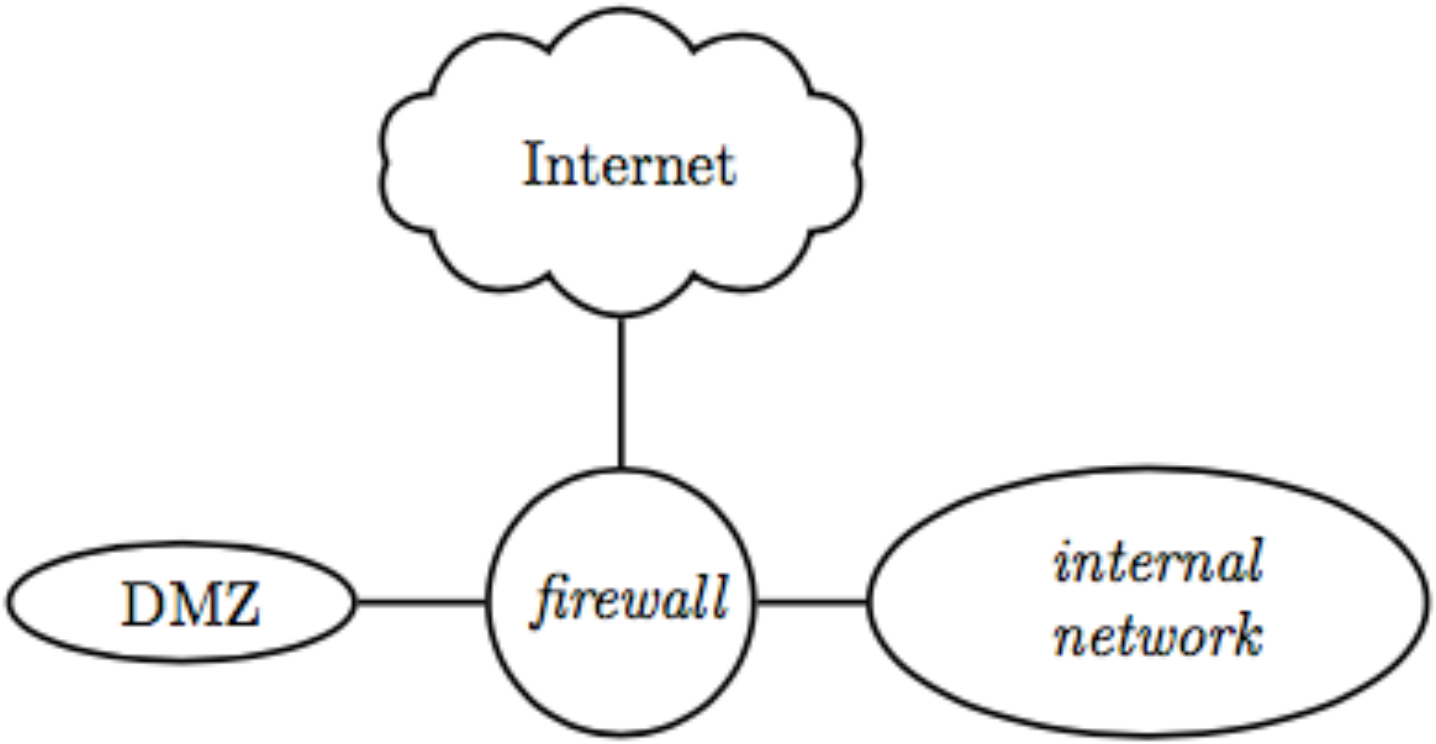
DMZ

- Portion of a network separating a completely internal network from an external one
  - Internal firewall separates DMZ, internal network
  - External firewall separates DMZ, external network
  - Internal firewall more restrictive than external one (usually)

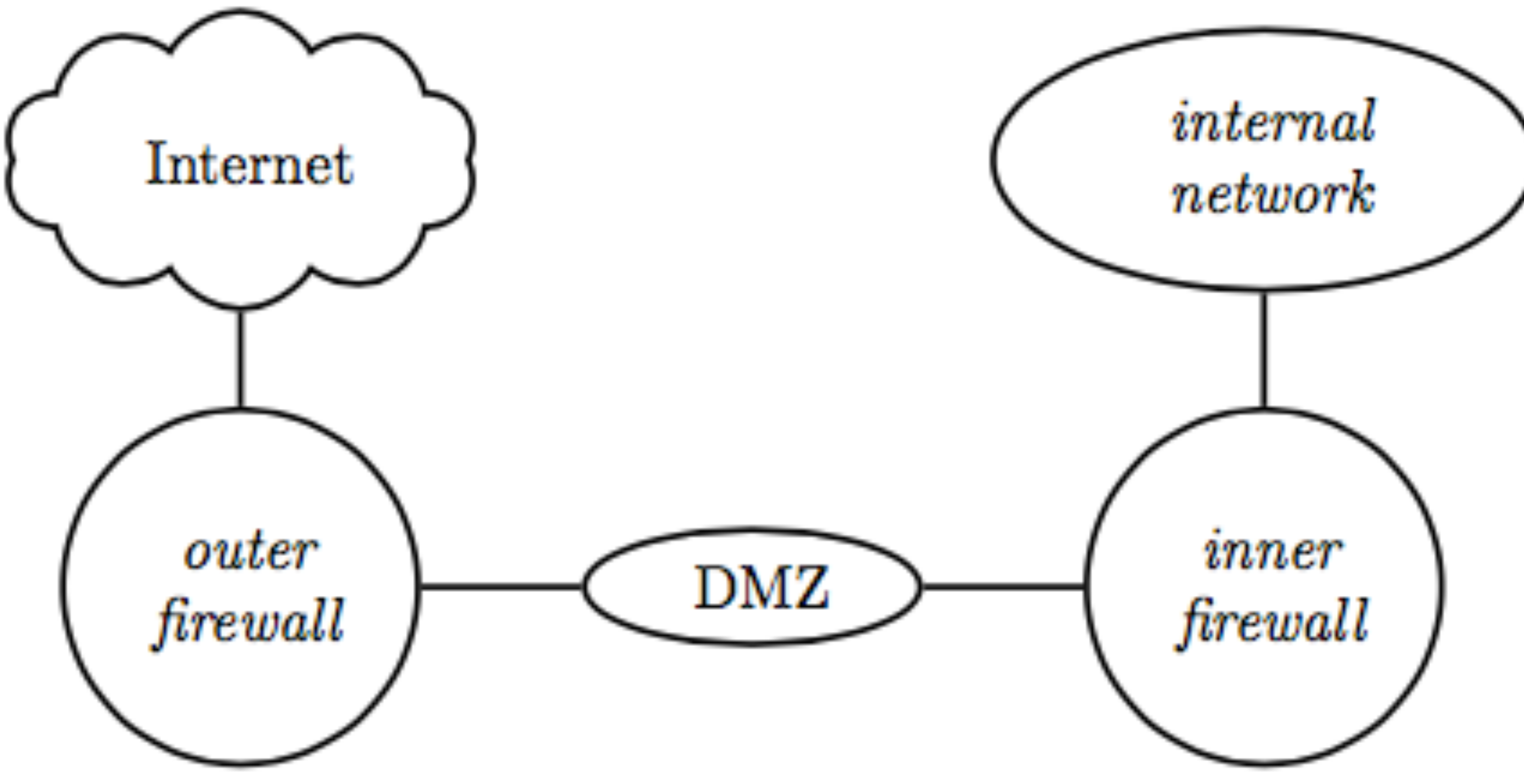
DMZ

- Idea: servers in DMZ serve as intermediaries
  - House externally visible web pages there
  - Email goes through a DMZ server
- If attacker compromises those systems, still must get through inner firewall to access company's secret

DMZ Configuration



DMZ Configuration



Confinement

- What is the problem?
- Isolation: virtual machines, sandboxes
- Detecting covert channels

Example Problem

- Server balances bank accounts for clients
- Server security issues:
  - Record correctly who used it
  - Send *only* balancing info to client
- Client security issues:
  - Log use correctly
  - Do not save or retransmit data client sends

Generalization

- Client sends request, data to server
- Server performs some function on data
- Server returns result to client
- Access controls:
  - Server must ensure the resources it accesses on behalf of client include *only* resources client is authorized to access
  - Server must ensure it does not reveal client's data to any entity not authorized to see the client's data

Confinement Problem

- Problem of preventing a server from leaking information that the user of the service considers confidential

Total Isolation

- Process cannot communicate with any other process
- Process cannot be observed

Impossible for this process to leak information

- Not practical as process uses observable resources such as CPU, secondary storage, networks, etc.

Example

- Processes p, q not allowed to communicate
  - But they share a file system!
- Communications protocol:
  - p sends a bit by creating a file called *0* or *1*, then a second file called *send*
    - p waits until *send* is deleted before repeating to send another bit
  - q waits until file *send* exists, then looks for file *0* or *1*; whichever exists is the bit
    - q then deletes *0*, *1*, and *send* and waits until *send* is recreated before repeating to read another bit

Covert Channel

- A path of communication not designed to be used for communication
- In example, file system is a (storage) covert channel

Rule of Transitive Confinement

- If p is confined to prevent leaking, and it invokes q, then q must be similarly confined to prevent leaking
- Rule: if a confined process invokes a second process, the second process must be as confined as the first

Lipner's Notes

- All processes can obtain rough idea of time
  - Read system clock or wall clock time
  - Determine number of instructions executed
- All processes can manipulate time
  - Wait some interval of wall clock time
  - Execute a set number of instructions, then block

Kocher's Attack

- This computes $x = a^z \bmod n$, where $z = z_0 \ldots z_{k-1}$

```
x := 1; atmp := a;
for i := 0 to k–1 do begin
  if zi = 1 then
     x := (x * atmp) mod n;
  atmp := (atmp * atmp) mod n;
end
result := x;
```

- Length of run time related to number of 1 bits in z

Isolation

- Present process with environment that appears to be a computer running only those processes being isolated
  - Process cannot access underlying computer system, any process(es) or resource(s) not part of that environment
  - A *virtual machine*
- Run process in environment that analyzes actions to determine if they leak information
  - Alters the interface between process(es) and computer

Virtual Machine

- Program that simulates hardware of a machine
  - Machine may be an existing, physical one or an abstract one
- Why?
  - Existing OSes do not need to be modified
    - Run under VMM, which enforces security policy
    - Effectively, VMM is a security kernel

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