Security Engineering

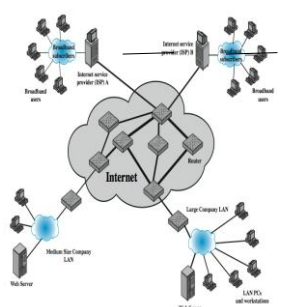
Lecture 17 – OS/VM Security
Fabio Massacci
(with W2K courtesy of D. Gollmann)

A misconception

- ***I don't need OS security because I consider smart sensors and***
  - they use machine-to-machine communication
  - they communicate either with wireless or power-lines
  - So once we secure the network we are done
- ***I don't need safety belts on my delivery van because***
  - we only deliver groceries door-to-door
  - we drive either on state roads or on country roads
  - So once we put brakes we are done

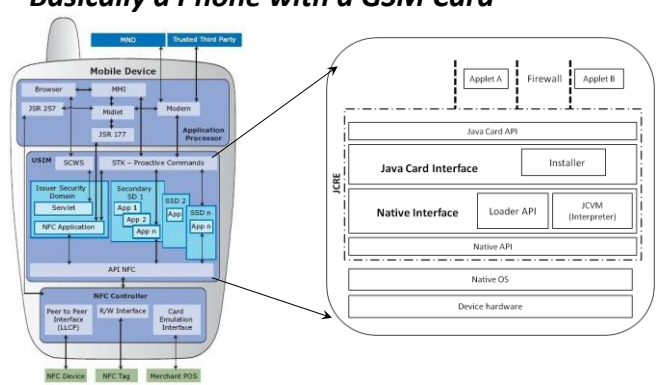
Some Misinterpreted Pictures..

- ***The picture is "evocative"***
  - but this is NOT the reality
- ***A "descriptive" picture would include all the different software and protocol stacks***
  - A MSc student in CS should know the actual reality…
  - And reason on what is really going on

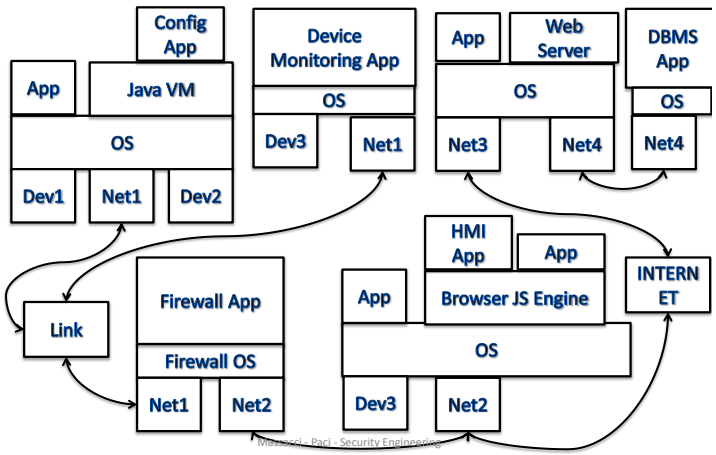


What is a smart sensor?

- ***Basically a Phone with a GSM Card***



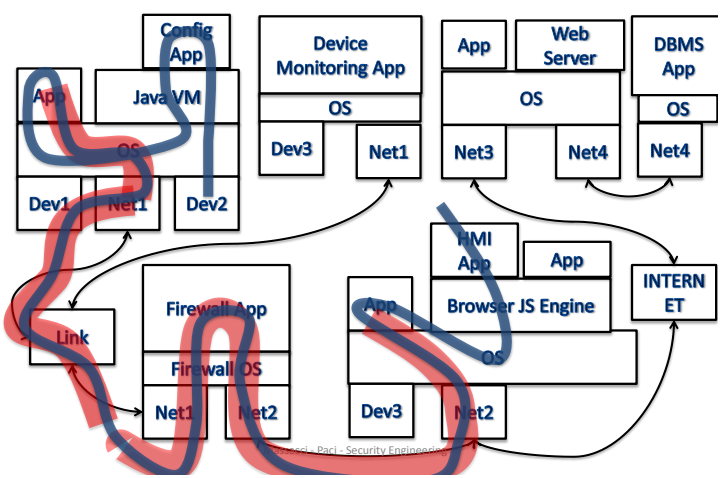
The Network…Actually



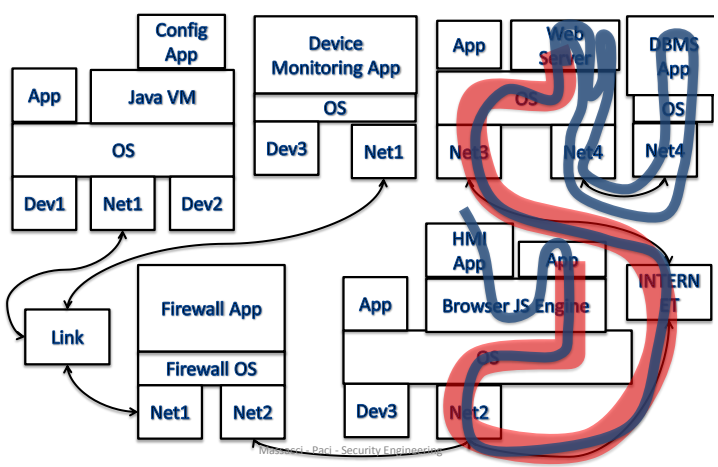
Some Security Technologies

- ***Transport Layer Security protocol, ver 1.0***
  - Confidentiality and data integrity between two communicating applications
  - Protect information transmitted between browsers and Web servers
  - Deployed in nearly every web browser
- ***IPSec authentication***
  - confidentiality, authentication, key management
- ***Where do we position them in the real picture?***

IPSEC+Configuration of Device



TLS+Selecting the Sensor on Server



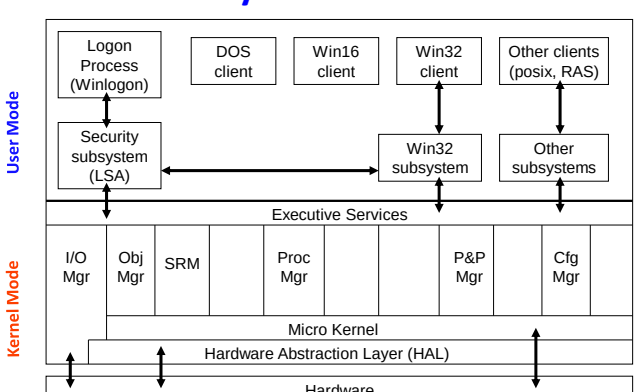
A Simple Model of the OS/VM

- ***A system is a collection of running processes and files.***
  - processes perform actions on behalf of a user
    - open, read, write files read, write, execute memory, etc.
  - files have access control lists dictating who can do users what
- ***Simple policy goals***
  - Integrity: processes running on behalf of user A shouldn't be able to corrupt the code, data, or files of user B nor interfere with the latter processes.
  - Availability: processes should eventually gain access to resources such as the CPU or disk.
  - Confidentiality: same as integrity (replace "corrupt"→ "read")
- ***More sophisticated goals***
  - Access control following a RBAC/MAC model

Windows 2000

- ***W2K is based on Windows NT***
  - most security features on NT also in W2K.
- ***W2K comes in various flavours***
  - Workstation, Server, Advanced Server and Datacenter.
  - The basics are the same in all cases, but the administration is different.
- ***In this lecture***
  - Workstation and Server

W2K System Architecture



Security Components

- ***Object Manager***
  - Manages objects, including files, folders, ports, processes and threads; is in charge of naming, maintaining security, allocating and disposing of objects.
- ***Security Reference Monitor (SRM)***
  - Validates access rights; compares a process' access token with an object's ACL and determines whether the requested access is granted; called by the Object Manager.
- ***Programs cannot access objects directly all accesses channelled through the O.S.***

Additional Security Objectives

- ***Authentication***
  - Single sign-on in the enterprise
  - Strong authentication
  - Active Directory (AD)
- ***Access Control***
  - Usage of security policies
  - Integrated security services
  - Delegation and scalability of administration
- ***Standards-based protocols for interoperability***
- ***Auditing services***

W2K Security Subsystem

- ***Authentication***
  - Active Directory (AD) Service – users, group policies
  - Kerberos (v5) – authenticates all W2K machines, and clients that support Kerberos authentication.
  - Secure Sockets Layer (SSL) – encrypted channel for authentication.
  - NTLM – protocol for logon to local user account; also supported in domain logons for older Windows machines.
- ***Access Control***
  - Local Security Authority (LSA) – the TCB: generates access tokens, manages local security policies, provides authentication for user logons.
  - Security Accounts Manager (SAM) – database of local users and accounts; used for local user authentication; stored locally on all non-domain controlled W2K machines.
  - Security Reference Monitor (SRM) – see previous slide

User Management (III)

- ***There are a number of predefined groups for a domain, mainly for management tasks***
- ***Administrators***
  - Users with rights to manage the system
- ***Account Operators***
  - Users with rights to manage user accounts.
- ***Server Operators***
  - Users with rights to manage servers.
- ***Users***
  - Normal users with accounts
- ***Guests***
  - Users without accounts who have restricted rights.
- ***etc***

Access Control

- ***Two mechanisms are used when a user attempts to access an object***
  - User Rights
  - Discretionary Access Control Lists (DACLs).
- ***All objects have Security Descriptors***
  - The SID of the user who owns the object, usually the creator of the object.
  - The DACL, which holds information about which users or groups can access the object. A DACL is a list of Access Control Entries (ACEs).
  - A System Access Control List (SACL) which defines the auditing policy for the object.

The Simple Way: User Rights

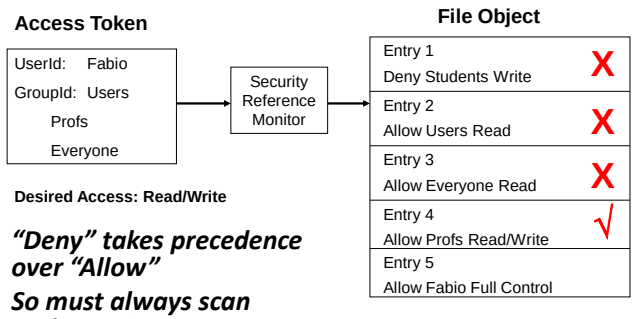
- ***Authorise users or groups to perform specific actions.***
  - Actually the SID associated with the user or group
  - Many possibilities and must be handled with care.

Right	Description	Default Groups
Act as part of the OS	Allows a user or group to run a process as a trusted part of the OS	None
Bypass traverse checking	Allows users/groups to traverse folders for which they have no access to allow access to a child folder to which they do have access.	Everyone Administrators Authenticated Users
Change the system time	Allows a user or group to set the system time of the computer	Administrators Server Operators Power Users

Potentially Dangerous Rights

- ***Some user rights have a high degree of risk associated with their possible misuse***
- ***Potentially dangerous rights must only be assigned to users/groups that actually need them.***
  - Act as part of the operating system
  - Create a token object
  - Add workstations to a domain
  - Back up files and directories
  - Change the system time
  - Debug programs
  - Increase scheduling priority
  - Increase quotas
  - Load and unload device drivers
- ***Try to answer why for each of them***

DACL (Intuition)



- ***"Deny" takes precedence over "Allow"***
- ***So must always scan entire DACL***

Auditing

- ***W2K auditing is an Administrator function***
  - can be assigned to an "Auditor" group
- ***Audit event categories include (can set audit success and/or fail)***
  - System events
  - Process Tracking
  - Privilege use
  - Policy change
  - Object access (i.e SACL)
  - Logon events
  - Directory Service access
- ***Security Log can be accessed via the Event Viewer.***
- ***Filtering options can be applied, logs can be saved to file, log sizes can be restricted, etc.***

Auditing (cont)

- ***Decisions***
  - what information needs to be collected.
  - what information does not need to be collected.
  - who should have access to the information.
- ***Actions***
  - Configure the system as appropriate to the environment.
  - Use a third-party log consolidation product if it is not practical to review all logs manually.
  - Review the information and, if necessary, act upon it.
- ***Important criterias***
  - What current Laws ask you to do?
  - What is useful for Computer Forensic?
  - Never store something you'll never look at (except for previous two exceptions)

What can go wrong?

- ***read/write/execute or change ACL of a file for which process doesn't have proper access.***
  - checkfileaccessagainstACL
- ***process writes (or reads) into memory of another process***
  - Isolate memory of each process (don't forget OS, network and device services etc. etc.)
- ***process pretends it is the OS and execute its codes***
  - maintain process ID and keep certain operations privileged
  - need some way to transition and avoid process transition back
- ***process never gives up the CPU***
  - force process to yield in some finite time
- ***process uses up all the memory or disk***
  - Enforce quotas
- ***OS is buggy … Oops.***

Security Attack on HBGary

- ***Casus Belli***
  - CEO Aaron Barr stated he would reveal Anonymous member.
- ***The beach head***
  - A custom written CMS application was exploited with SQL injection and the usernames/passwords were dumped from the users table.
  - The passwords were hashed with MD5 but not salted so simple rainbow tables cracked some of the passwords.
    - The CEO and COO had passwords were six lower-case letters and two numbers.
    - Now the attackers had access to the CMS plus whoever re-used these passwords

Security Attack on HBGary

- ***A first inroad***
  - One password cracked was reused and had SSH non-root access to support.hbgary.com.
  - Elevated to root access using a known local privilege-escalation vulnerability of an unpatched.
  - Gigabytes of research info removed
- ***A second inroad***
  - Same passwords used in Google, Twitter, and LinkedIn. CEO's was admin for Google Apps Mail
  - By resetting users passwords, could gain access.
- ***Finally***
  - One accounts, "Greg Hoglund", disclosed two potential root account passwords to a rootkit.com server. Also revealed that a Nokia employee had SSH access to that server.
  - ssh as root not allowed so attacker impersonated "Greg Hoglund" and wrote to Nokia employee to get ssh onto server
  - Once on server was able to elevate to root
- ***They defaced the server…***

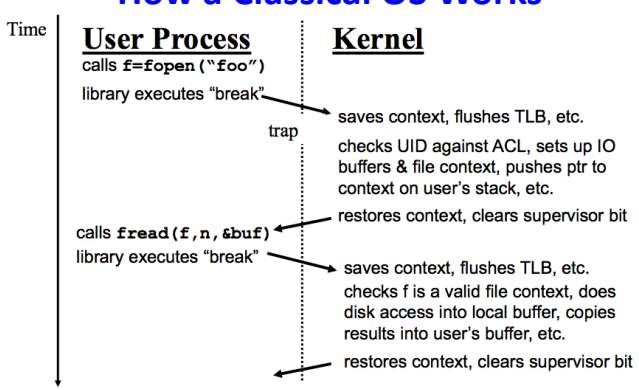
What an OS should have?

- ***reliable access to information about what the App is about to do***
  - what instruction is it about to execute?
  - Which data is going do be read ot written
- ***ability to "stop" the application***
  - can't stop a program running on another machine that you don't control
  - really, stopping isn't necessary, but transition to a "good" state.
- ***Ability to protect the OS's state and code from tampering.***
  - key reason why a kernel's data structures and code aren't accessible by user code.
- ***More and above all that → low overhead.***

How does it work at Hw level?

- ***Translation Lookaside Buffer (TLB)***
  - provides an inexpensive check for each memory access.
  - mapsvirtualaddresstophysicaladdress
    - small, fully associative cache (8-10 entries) – cache miss triggers a trap
    - granularity of map is a page (4-8KB)
- ***Distinct user and supervisor modes***
  - certain operations (e.g., reload TLB, device access) require supervisor bit is set
  - Invalid operations cause a trap
- ***Setsupervisor bit and transfer control back to OS routine.***
  - Timer triggers a trap for preemption and avoids hijacking

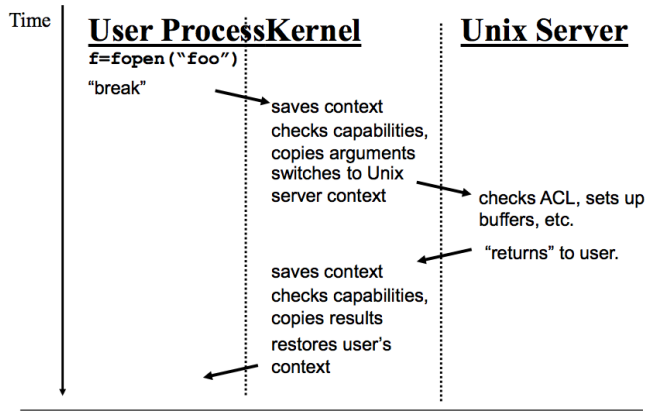
How a Classical OS Works



MicroKernels

- ***The smaller the VMM/Sandbox the better***
  - Increase Flexibility,
  - Minimize the TCB
- ***A big push for microkernels***
  - Mach, Spring, etc.
- ***Only put bare minimum into the kernel.***
  - context switching code, TLB management
  - trap and interrupt handling device access
- ***Run everything else as a process.***
  - file systems networking protocols page replacement algorithm
- ***Component Sub-systems communicate via remote procedure call (RPC)***

How Micro-Kernels works



Performance trumps…

- ***Claim was that flexibility and increased assurance would win***
  - But performance overheads were non trivial
  - Many PhD's on minimizing overheads of communication
  - Even highly optimized implementations of RPC cost 2/3 orders of magnitude more than a procedure call.
- ***Result: micro-kernel won't fly***
  - Some embedded or specialized kernels (e.g., Exokernel)
- ***Windows, Linux, Solaris***
  - continue the monolithic tradition.
  - and continue to grow for performance reasons (e.g., GUI) and for functionality gains (e.g., specialized file systems.)
- ***Mac OS X, Free BSD***
  - Originally based on Mach, but nowadays
    - "The OS X kernel environment includes the Mach kernel, BSD, the I/O Kit, file systems, and networking components."
- ***VMware***
  - achieves multiple personalities but has monolithic personalities sitting on top

The curse of performance

- ***If performance was not an issue an OS could:***
  - examine the entire history and the entire machine state to decide whether or not to allow an instruction.
  - perform an arbitrary computation to decide whether or not to allow a transition.
  - Use a distinct instruction set (and processor) from the program
- ***In practice, most systems must***
  - keep a small piece of state to track mostr recent history
  - only look at labels on the transitions
  - have small and few labels
  - perform simple tests
  - use (almost) the same instruction set
- ***Otherwise, the overheads would be overwhelming.***
- ***So policies are practically limited by the vocabulary of labels, the complexity of the tests, the state maintained by the OS/VM, and the potentially different instructions***

Two Alternative Protection models

- ***Sandboxing***
  - Does not emulate computer's hardware
  - Alters interface between computer, process
  - Requires only software support
- ***Virtual machines***
  - Emulate computer's hardware
  - "Guest" entity cannot access underlying computer system
  - Requires absolutely hardware support

Sandboxes

- ***Environment in which actions of process are restricted according to security policy***
  - Program to be executed is not altered,
  - Implementation of "Interface" instructions with devices is changed
    - Can add extra security-checking mechanisms to libraries, kernel, drivers, etc.
  - Similar to debuggers, profilers that add breakpoints
  - Example → JavaVM, Browsers, Android etc.
- ***Sometimes can modify program or process to be executed***
  - Add code to do extra checks (memory access, etc.) as program runs (software fault isolation)
    - Not truly sandboxing in this case → in-line monitor
  - Example → Software Fault Isolation

Virtual Machine

- ***A program that simulates hardware of computer system and reports results back to Application***
  - Classical OS is essentially the first "virtualization" of the physical hardware
- ***Virtual machine monitor (VMM, "hypervisor") provides VM on which conventional OS can run***
  - Each VM is one subject;
  - VMM doesn't worry about processes running inside each VM
    - up to the VM manager to make sure they are properly secure
  - VMM mediates all interactions of VM with resources or other VMs

Additional Readings

- ***Gollmann – Computer Security***
  - Ch. 8 – Operating Systems
  - Ch. 9 – Databases
- ***Mac OS X kernel Information***
  - https://developer.apple.com/library/mac/documentation/Darwin/Conceptual/KernelProgramming/Architecture/Architecture.html
- ***On exploiting execution path to gain control of Applications and OS***
  - Tutorial by Lucas Davi, ESWeek 2015