UNIVERSITY OF TRENTO

Security Engineering

Lecture 17 - OS/VM Security

Fabio Massacci

UNIVERSITY OF TRENTO

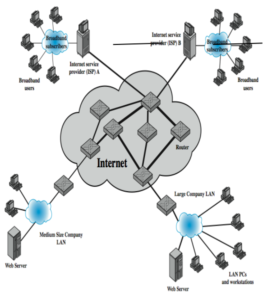
A misconception

- 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

UNIVERSITY OF TRENTO

Some Misinterpreted Pictures..

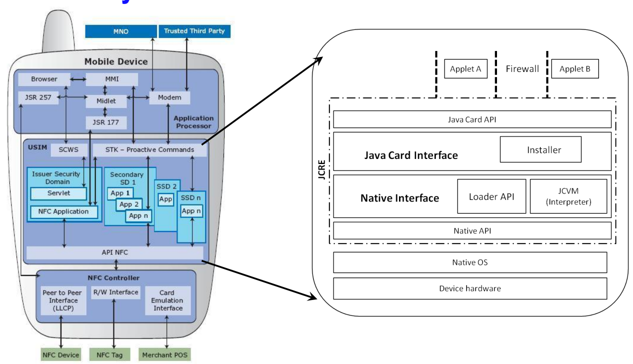
- The picture is "evocative"
  - but this is NOT the reality
- A "descriptive" picture would include all the different software and protocol stacks
  - A MSc student in CS should know the actual reality…
  - And reason on what is really going on



UNIVERSITY OF TRENTO

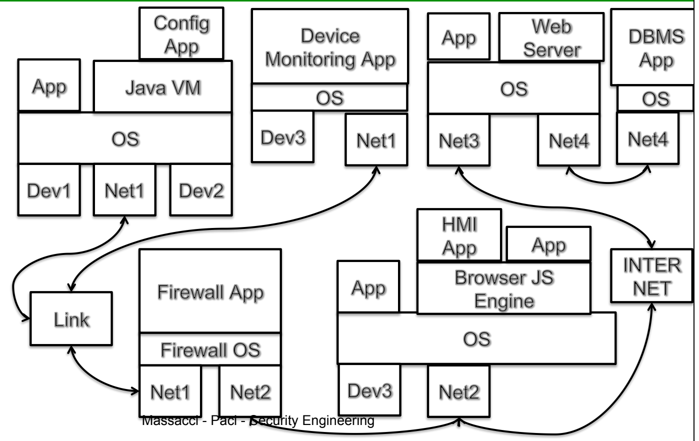
What is a smart sensor?

- Basically a Phone with a GSM Card



The Network…Actually

UNIVERSITY OF TRENTO



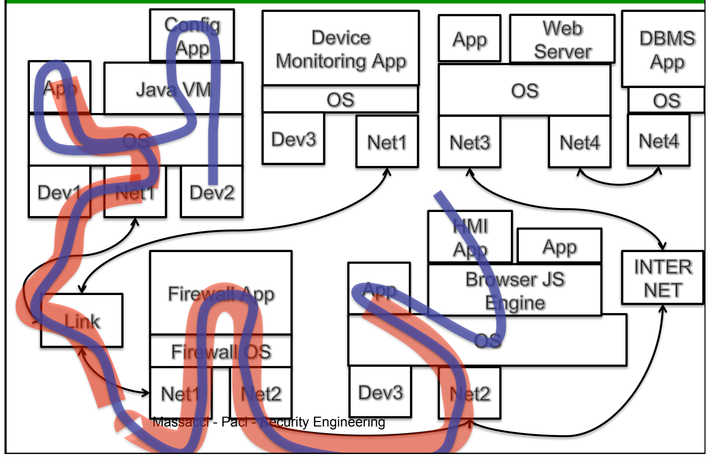
Some Security Technologies

UNIVERSITY OF TRENTO

- **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?**

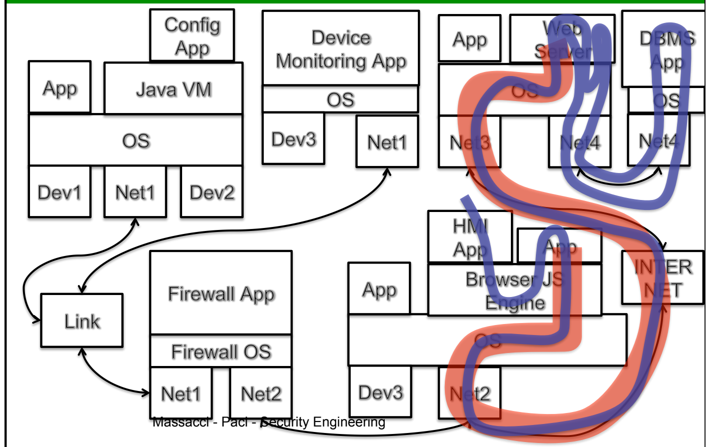
IPSEC+Configuration of Device

UNIVERSITY OF TRENTO



TLS+Selecting the Sensor on Server

UNIVERSITY OF TRENTO



A Simple Model of the OS/VM

UNIVERSITY OF TRENTO

- 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

What can go wrong?

UNIVERSITY OF TRENTO

- 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 or hardware is buggy ... Oops.

What an OS should have?

UNIVERSITY OF TRENTO

- 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.

The curse of performance

UNIVERSITY OF TRENTO

- 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

UNIVERSITY OF TRENTO

- 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

UNIVERSITY OF TRENTO

- 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
- 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

UNIVERSITY OF TRENTO

- 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

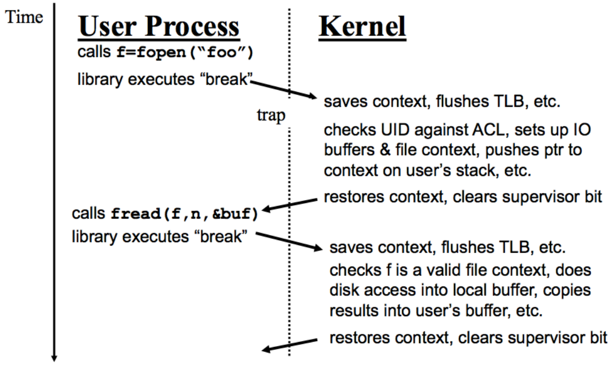
Hardware Support for OS/VM

UNIVERSITY OF TRENTO

- 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

UNIVERSITY OF TRENTO



Massacci - Paci - Security Engineering

MicroKernels

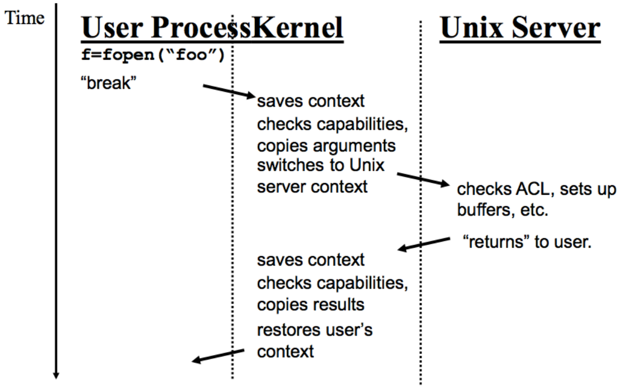
UNIVERSITY OF TRENTO

- **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)**

Massacci - Paci - Security Engineering

How Micro-Kernels works

UNIVERSITY OF TRENTO



Massacci - Paci - Security Engineering

Performance trumps…

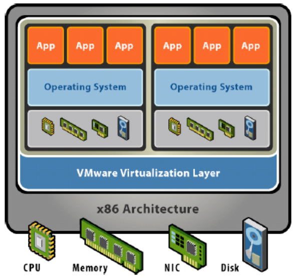
UNIVERSITY OF TRENTO

- **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**
- **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, some embedded or specialized kernels (e.g., Exokernel)**
  - exceptions.
- **VMware**
  - achieves multiple personalities but has monolithic personalities sitting on top
- **What about cloud architectures?**

Massacci - Paci - Security Engineering

Typical “Cloud” Scenarios

UNIVERSITY OF TRENTO



vmware Xen Parallels

- **Running one or more applications not supported by host OS**
  - A virtual machine running required guest OS could allow the desired applications to be run
- **Evaluating an alternate operating system**
  - The new OS could be run within a VM
- **Server virtualization**
  - Multiple virtual servers could be run on a single physical server, in order to more fully utilize the hardware resources of the physical server.
- **Duplicating specific environments**
  - A virtual machine could be duplicated and installed on multiple hosts.
- **Creating a protected environment**
  - If guest OS running on a VM becomes infected with malware, host operating system's exposure may be limited (depends on configuration of virtualization software)

Source: Wikipedia, VMWare

Reasons for Cloud Virtualization

UNIVERSITY OF TRENTO

- **Server consolidation (Physical-to-Virtual (P2V) transformation)**
  - many small physical servers → one larger physical server, to increase utilization of hw
  - The large server can "host" many such "guest" virtual machines
- **Inspection and isolation**
  - A virtual machine can be more easily controlled and inspected from outside than a physical one, and its configuration is more flexible.
- **Provisioning and relocation**
  - A new virtual machine can be provisioned as needed without the need for an up-front hardware purchase.
  - a virtual machine can easily be relocated from one physical machine to another as needed.
- **Disaster recovery scenarios**
  - Because of easy relocation
  - ONLY work if you have more machines in different locations. If you only have one big server won't work

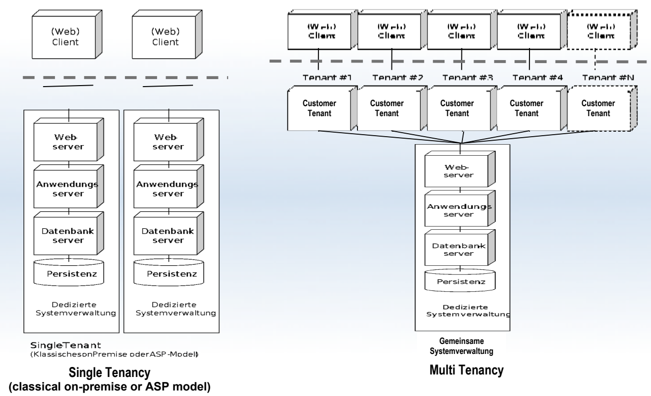
Cloud Architectural Solutions

UNIVERSITY OF TRENTO

- **SaaS (Software as a Service)**
  - A provider licenses an application to customers for use as a service on demand.
  - vendors host application on own web servers or download the application to consumer device, disabling it after contract expires.
- **PaaS (Platform as a service)**
  - delivery of computing platform & solution stack as a service.
  - facilitates deployment of applications without cost & complexity o buying and managing hardware & software layers.
  - Environment supports lifecycle for building & running applications
- **IaaS (Infrastructure as a Service)**
  - delivery of computer infrastructure as a service typically a virtualized environment managed in an integrated and efficient way.
  - Offers computing as a service billed on a utility basis and amount of resources consumed
- So we would expect a lots of isolation + virtualization…

From ASP to Multi-Tenancy

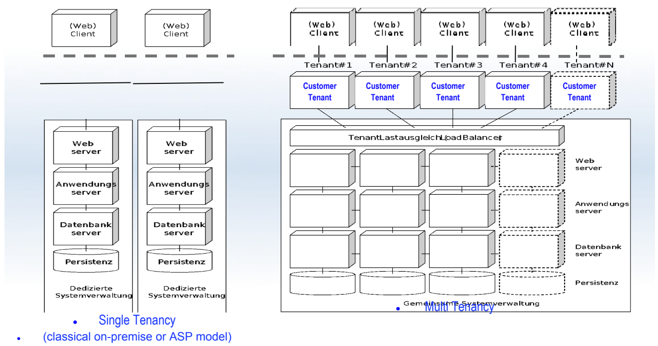
UNIVERSITY OF TRENTO



Source: SAP

Efficient & Sclabable Multi-Tenancy

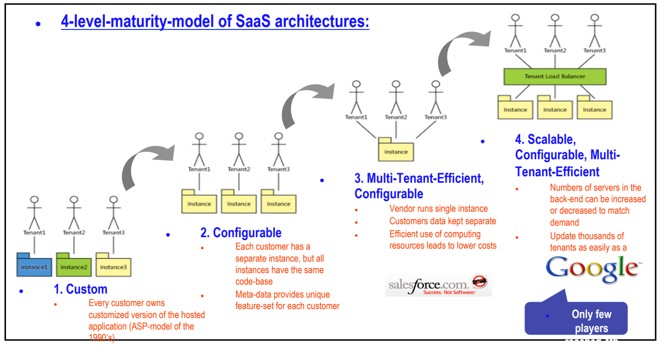
UNIVERSITY OF TRENTO



Source: SAP

The less isolation the "better"…

UNIVERSITY OF TRENTO



[MSDN, F. Chong and G. Carraro, "Architecture Strategies for Catching the Long Tail", http://msdn.microsoft.com/en-us/library/aa479069.aspx, April 2006.]

Performance wins again

UNIVERSITY OF TRENTO

- **The hit of crossing the kernel/OS boundary:**
  - Original Apache implementation forked a process to run each CGI:
  - Could attenuate file access for sub-process
  - protected memory/data of server from rogue script
    - Very close to least privilege
- **Too expensive for**
  - a small script (fork, exec, copy data to/from the server process), etc.
  - if this is repeated millions or billions of times…
    - can have more hardware but hardware don't scale equally well than clients
    - and you started all that to avoid having as much hardware as clients…
- **current push is to run the scripts in the server.**
  - See Node.JS raison d'etre…
  - Throw out least privilege
- **Similar situation with DBs, web browsers, file systems, etc.**

Additional readings

UNIVERSITY OF TRENTO

- **Gollmann – Computer Security**
  - Ch. 8 – Operating Systems
  - Ch. 9 – Databases
- **NIST Guide on Hypervisor**
  - csrc.nist.gov/publications/nistpubs/800-125/SP800-125-final.pdf
- **Search Google for DataCenter Security**
  - http://www.youtube.com/watch?v=1SCZzgfdTBo

Massacci - Paci - Security Engineering