Operating System Security Lecture 5

Virtualization, Sandboxing, and Emulation

Roel Verdult

Institute for Computing and Information Sciences
Radboud University Nijmegen, The Netherlands
✉ *rverdult@cs.ru.nl* / 🖱 *www.cs.ru.nl/~rverdult*

Introduction to VM's

- Virtualization (ring 0)
  - Hardware oriented
    - Native (VSphere, XEN, Hyper-V)
    - Host based (VMware, Virtualbox, Parallels)
  - Operating-System level (chroot, jails, Vserver)
    - Virtual private servers (VPS)
- Sandboxing (ring 3)
  - Application based (APP-V, App Sandbox, AppArmor)
  - Compiler based (Java, .NET, javascript)
- Emulation (ring 3)
  - Game consoles (Snes9x, Nestopia, Fusion)
  - Qemu (x86-64, ARM, MIPS)
  - Minemu (Dynamic taint analysis)

Virtualization (ring 0)

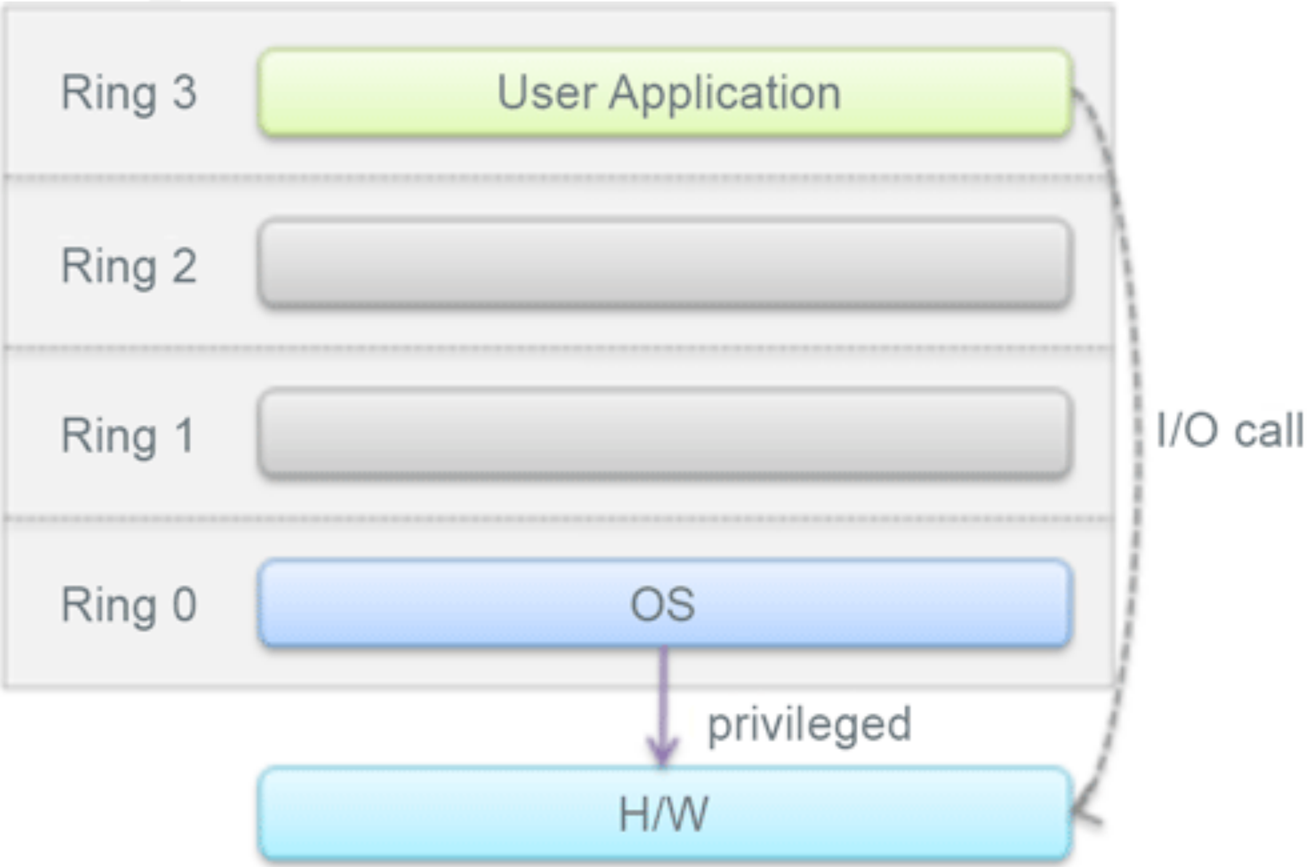
- Hardware oriented
  - Operating-system thinks it is running on and interacting with its own hardware
  - Abstracts the hardware peripherals from the operating-system

- Operating-System level
  - Makes the subsystem thinks it is running in its own operating-system
  - Abstracts the services and kernel from an application

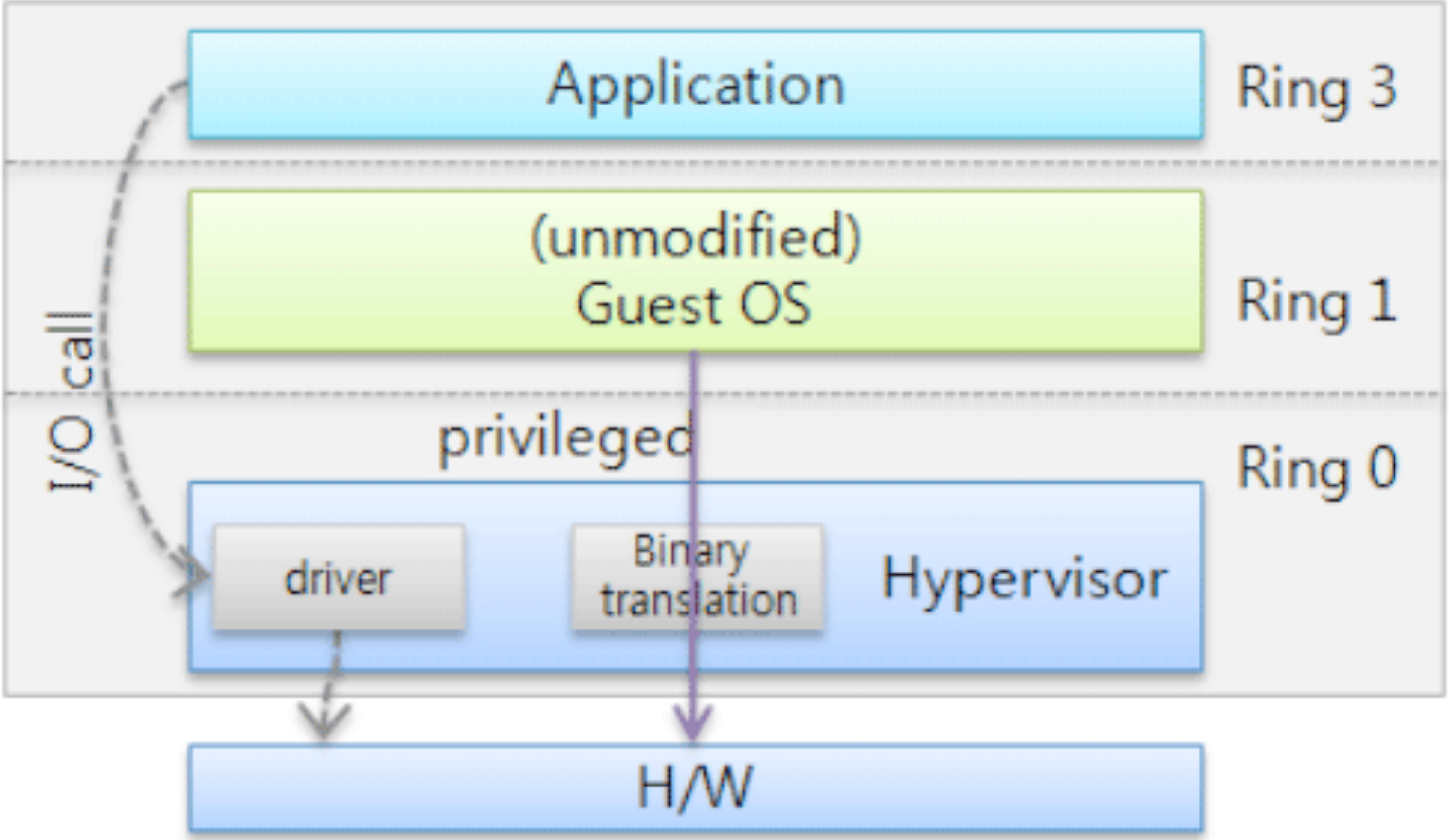
Native Hardware Virtualization

- Native (bare-metal) hypervisors
  - Run directly on the host's hardware to control the hardware and to manage guest operating systems
  - No reliance on an underlying OS
  - A guest operating system runs as a process on the host
  - IBM mainframes used native hypervisors in the 1960s (CP-40 IBM S360/S370)
  - Nowadays, many more alternatives
    - VSphere, XEN, Hyper-V

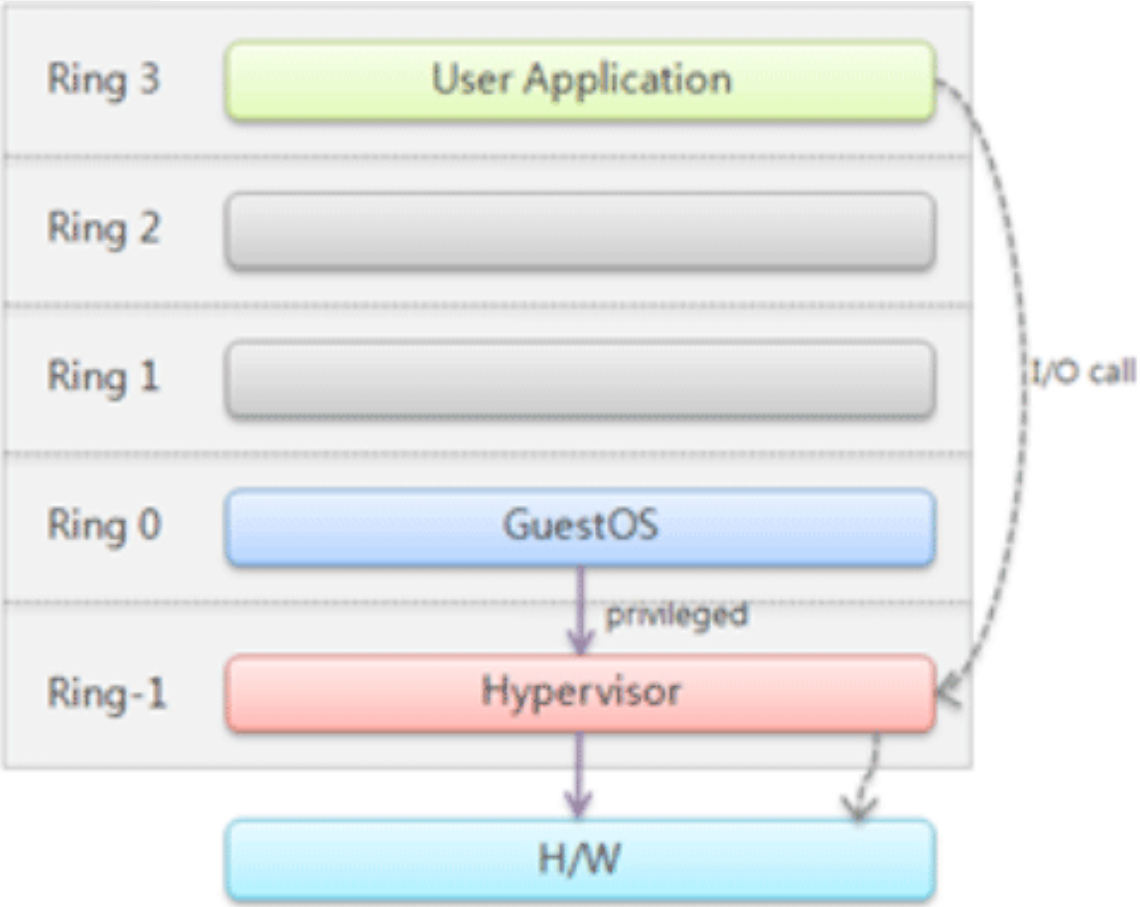
x86-64 Privileged Architecture



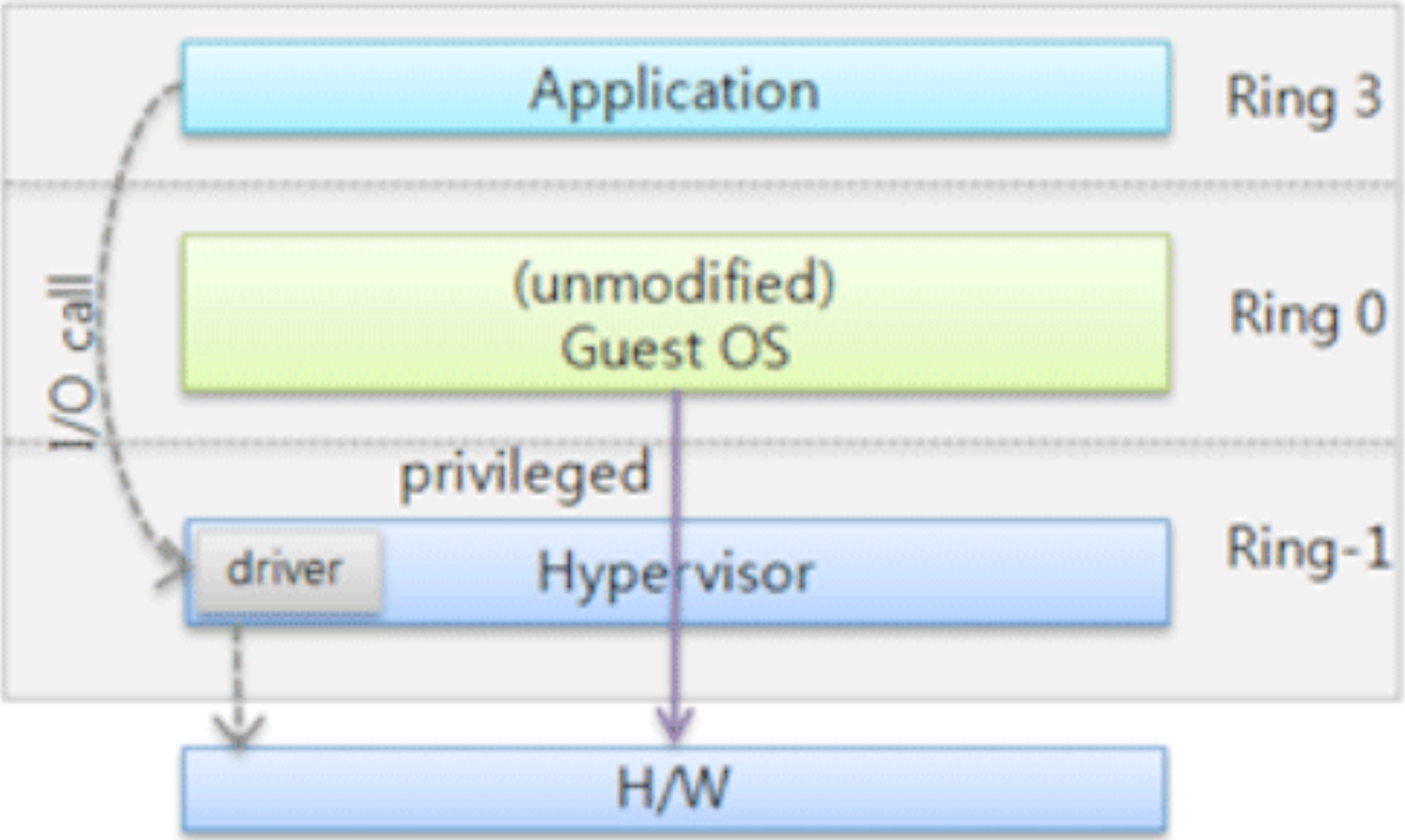
Full Virtualization



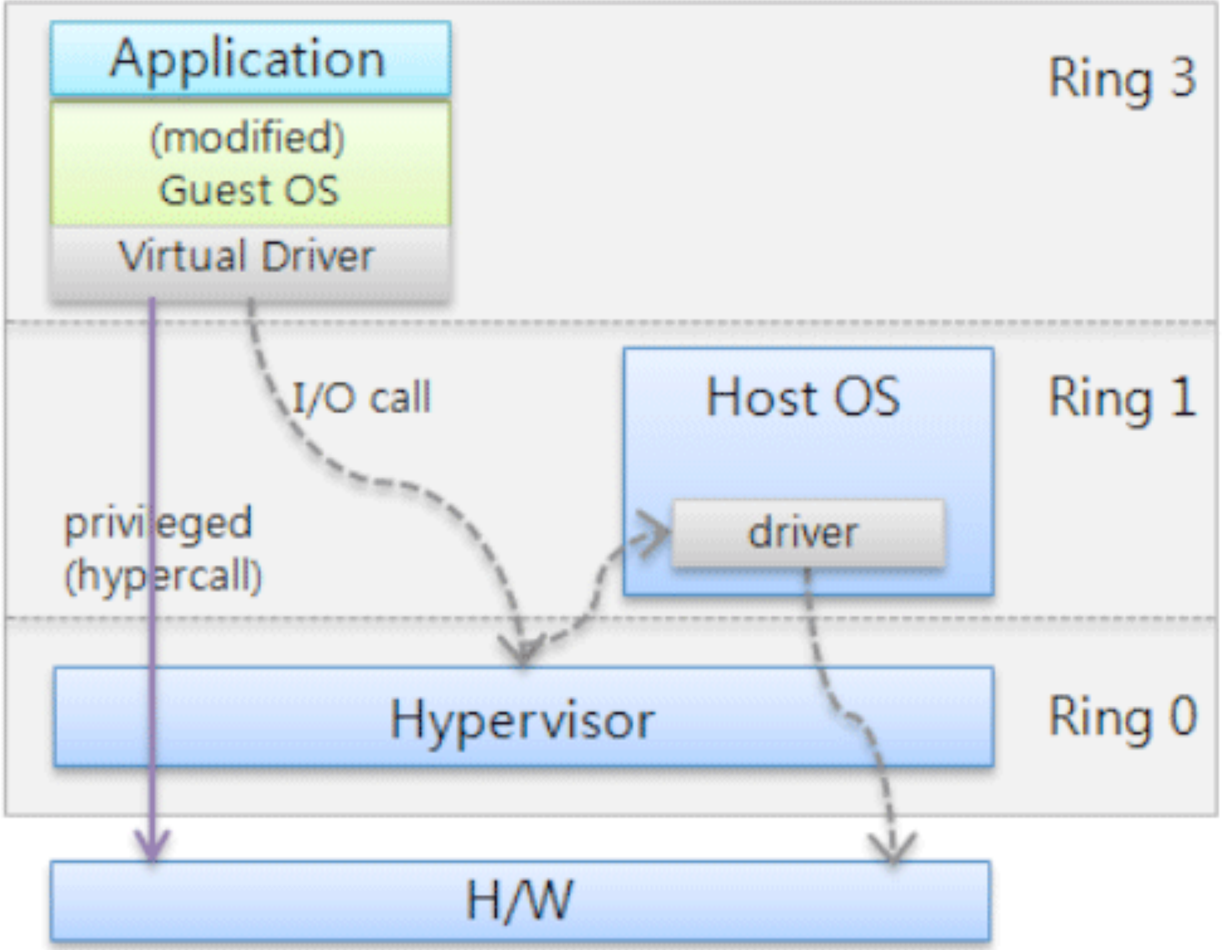
HW-assisted Virtualization



Intel VT-x Full Virtualization



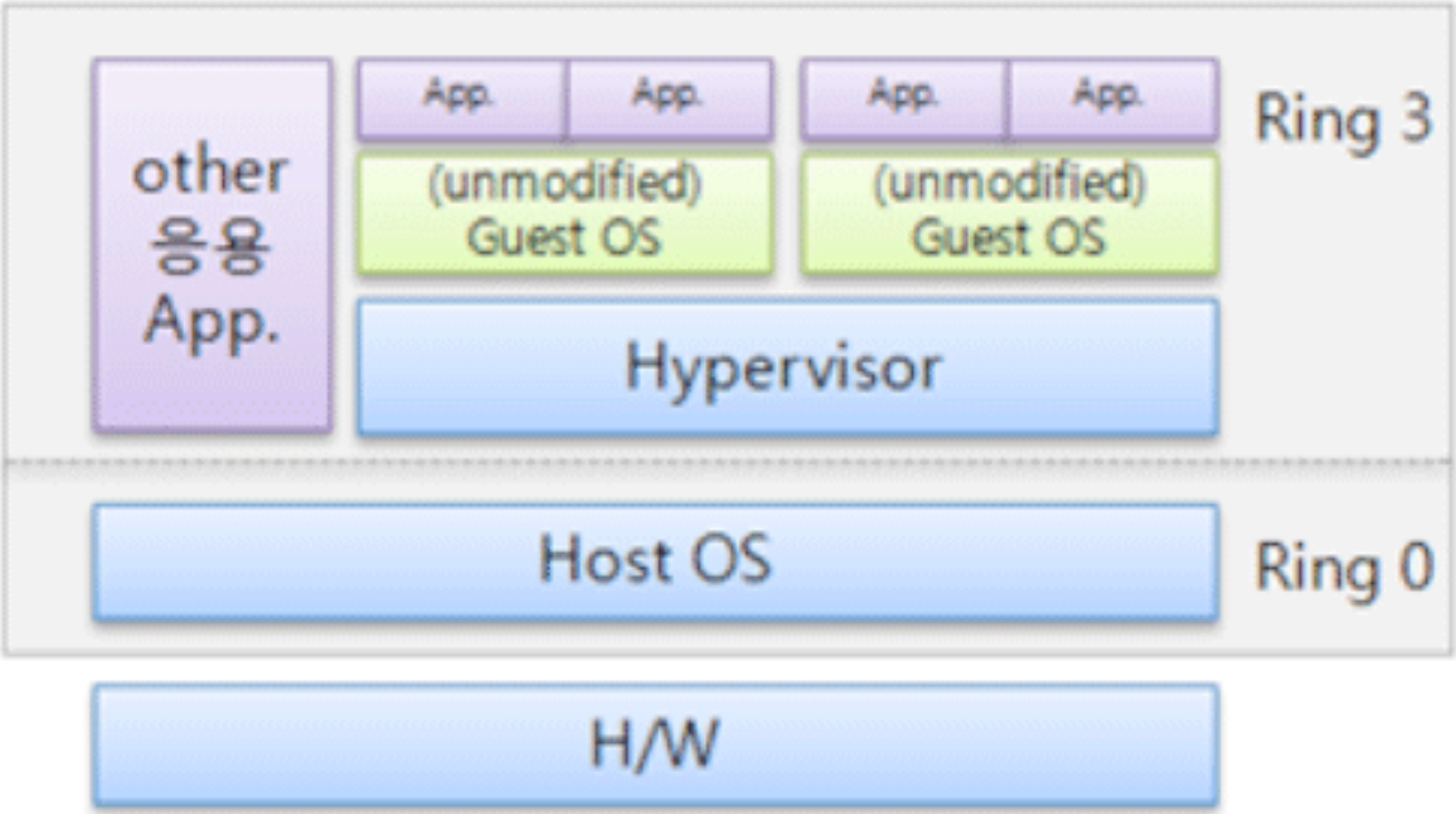
Para Virtualization (e.g. XEN)



Host based virtualization

- Relies on an underlying operating-system
- Hardware abstraction implicates that
  - The OS handles the peripherals and is responsible for the hardware drivers
  - The security environment is less controlled because of the reliance on the underlying OS
  - There is no direct access to hardware, which increases resource overhead of the guest OS
- Popular examples
  - WMware, Parallels, VirtualBox, Windows Virtual PC

Host based virtualization



OS level Virtualization

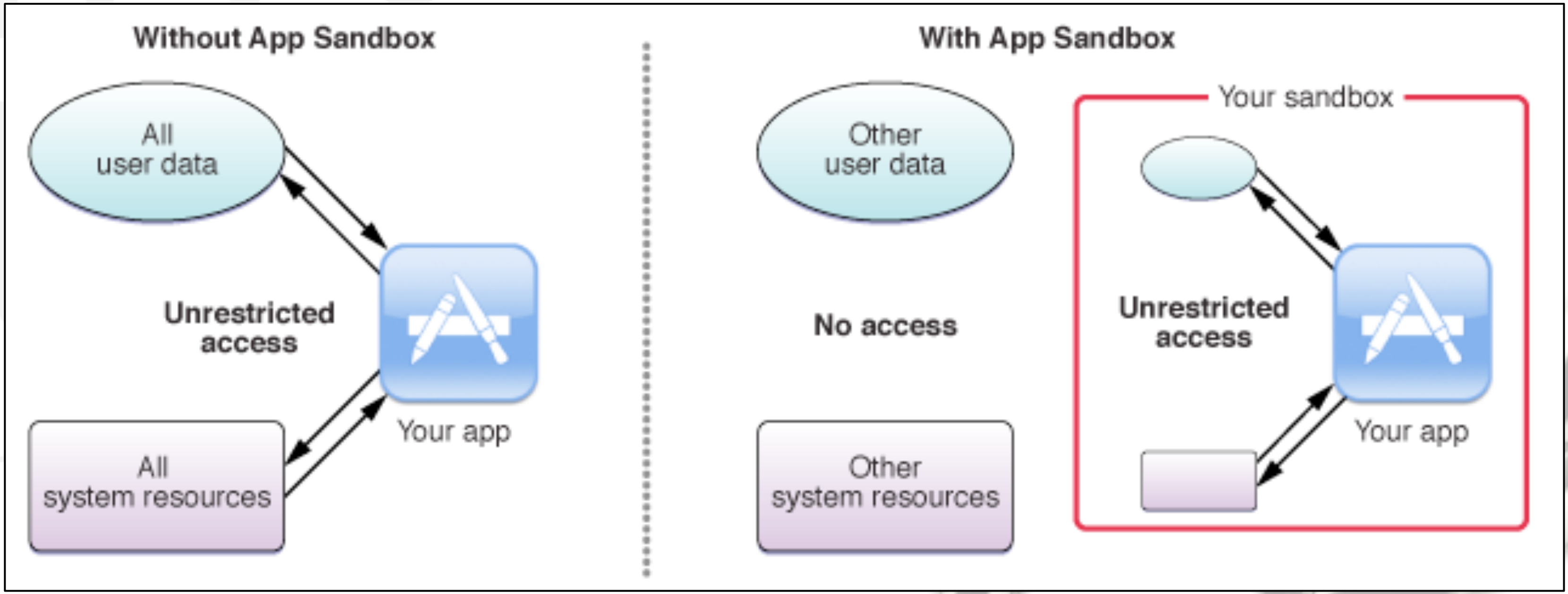
- Chroot
  - System call that virtualizes the file system. It basically changes the “root” folder for a process
- FreeBSD jails / Linux Vserver
  - System calls / kernel patch
  - Virtualizing file system
  - Resource limits (CPU, Memory)
  - Networking subsystem
  - No guest OS, one kernel for virtualized instances
  - Used by many webhosting companies that offer “cloud resources”, Virtual Private Servers (VPS) and web-based application services.

Sandboxing (ring 3)

- Apple App Sandbox
  - Removes most capabilities for interacting with the operating system
- Linux AppArmor
  - Applies security profiles per application which define what system resources individual applications can access, and with what privileges
  - By default installed and loaded on serveral mainstream Linux distributions (Ubuntu, openSUSE).
- Microsoft APP-V
  - Synchronizes a locally installed application and environment with a remote virtualized instance

Apple App Sandbox

- Removes most capabilities for interacting with user data and system resources.
- Avalable in iOS and OS X



Linux AppArmor Example

- **/etc/apparmor.d/bin.ping**

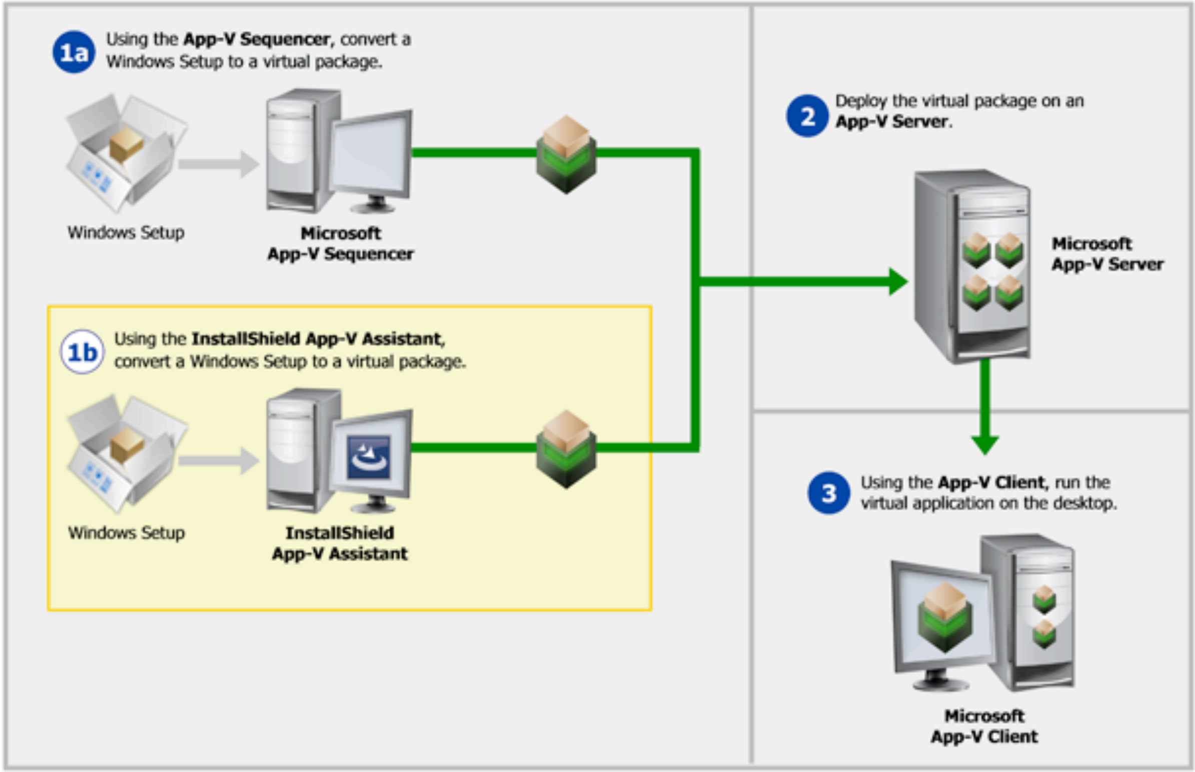
```
#include <tunables/global>
/bin/ping flags=(complain) {
  #include <abstractions/base>
  #include <abstractions/consoles>
  #include <abstractions/nameservice>

  capability net_raw,
  capability setuid,
  network inet raw,

  /bin/ping mixr,
  /etc/modules.conf r,
}
```

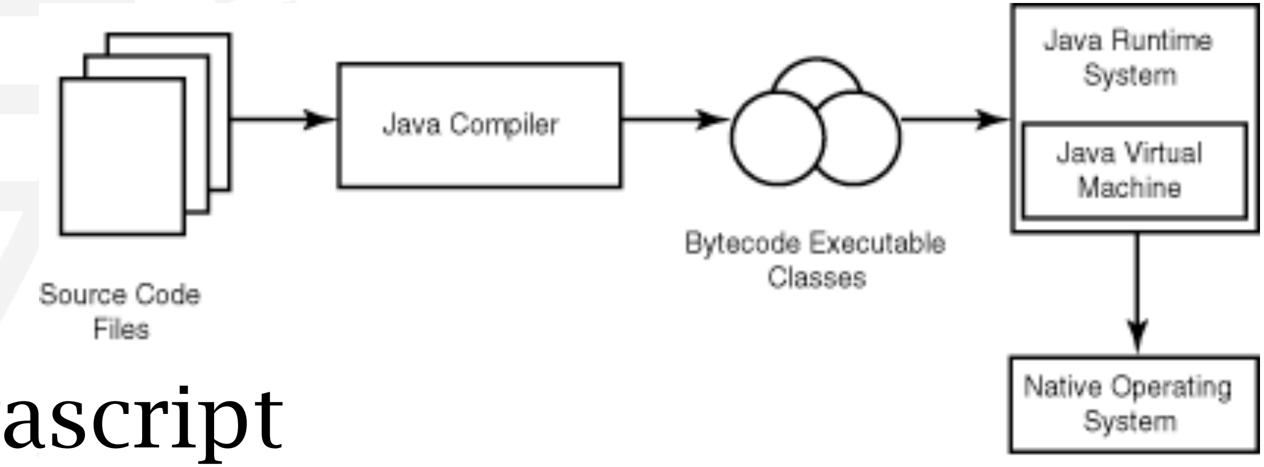
Microsoft APP-V

- Hybrid
  - Not remote
  - Not local
- Deployed & sync’ed to a remote instance


1a Using the App-V Sequencer, convert a Windows Setup to a virtual package.
Windows Setup
Microsoft App-V Sequencer
1b Using the InstallShield App-V Assistant, convert a Windows Setup to a virtual package.
Windows Setup
InstallShield App-V Assistant
2 Deploy the virtual package on an App-V Server.
Microsoft App-V Server
3 Using the App-V Client, run the virtual application on the desktop.
Microsoft App-V Client

Compiler sandboxing (ring 3)

- Java / .NET
  - Just-in-time compilation (JIT)



- Javascript
  - Browser integration
  - Eval() statements

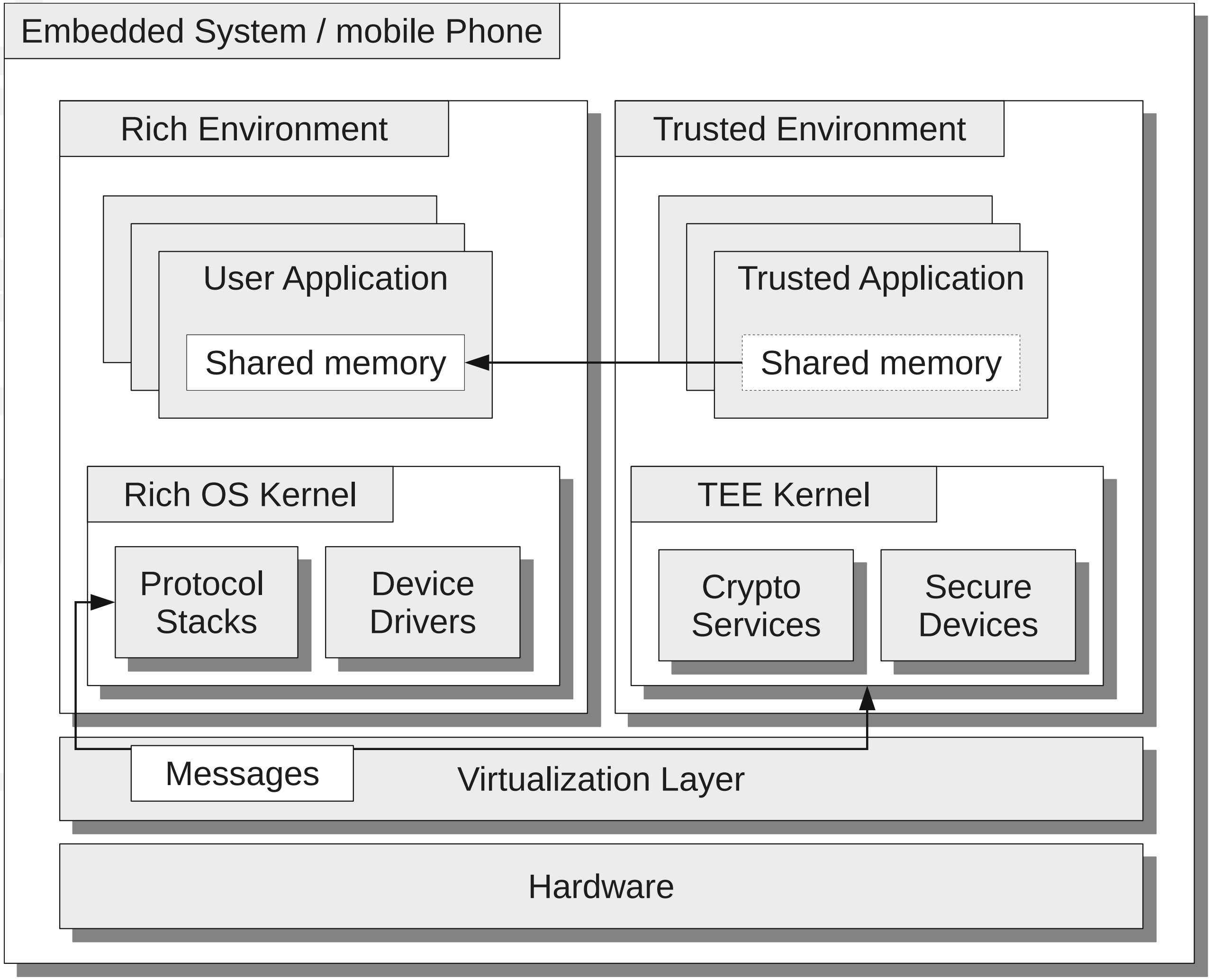
Emulation (ring 3)

- Malware analysis
- Reverse Engineering
- Driver debugging
- Forensics
  - Record / replay execution
  - Taint tracking
  - Symbolic execution

VM vulnerabilities

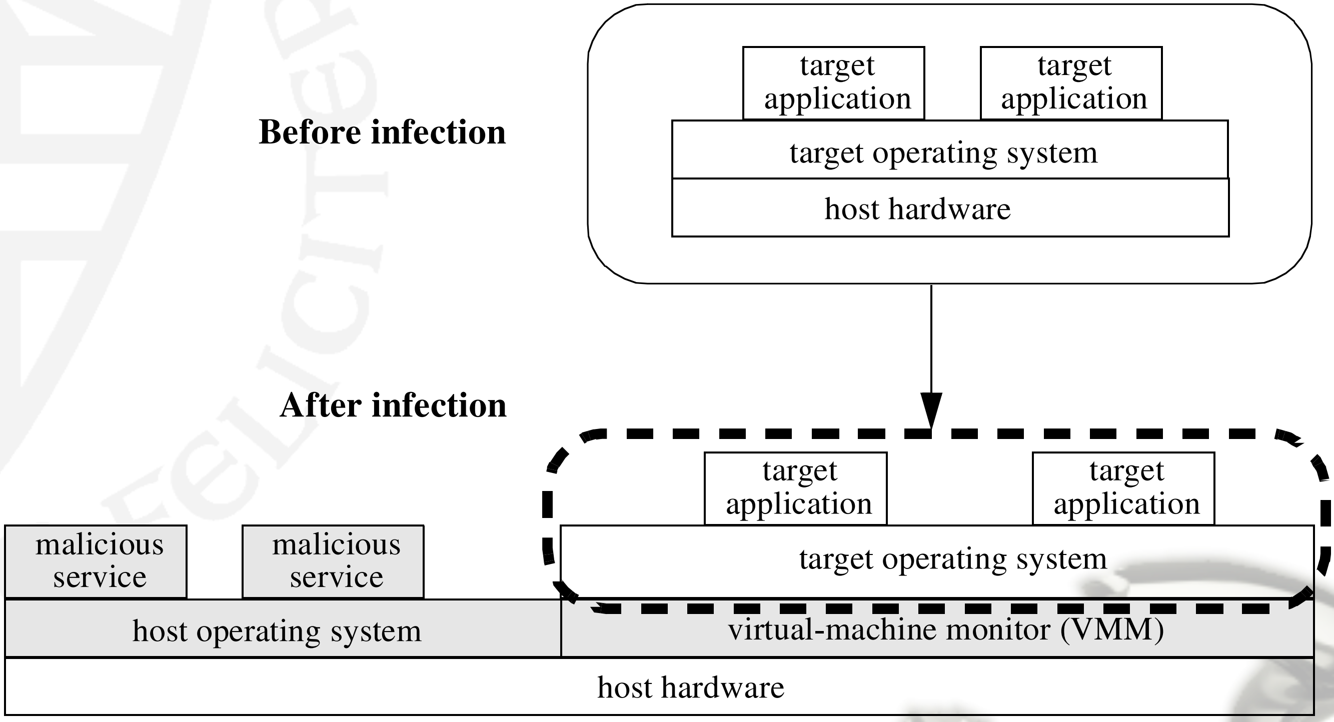
- Virtualization
  - Hardware oriented attacks
  - Management interface exploits
  - Break out of jail attacks (VM Escape)
- Sandboxing
  - Application privilege escalation
  - JIT Spraying
  - Untrusted native code execution

Cache based timing attacks

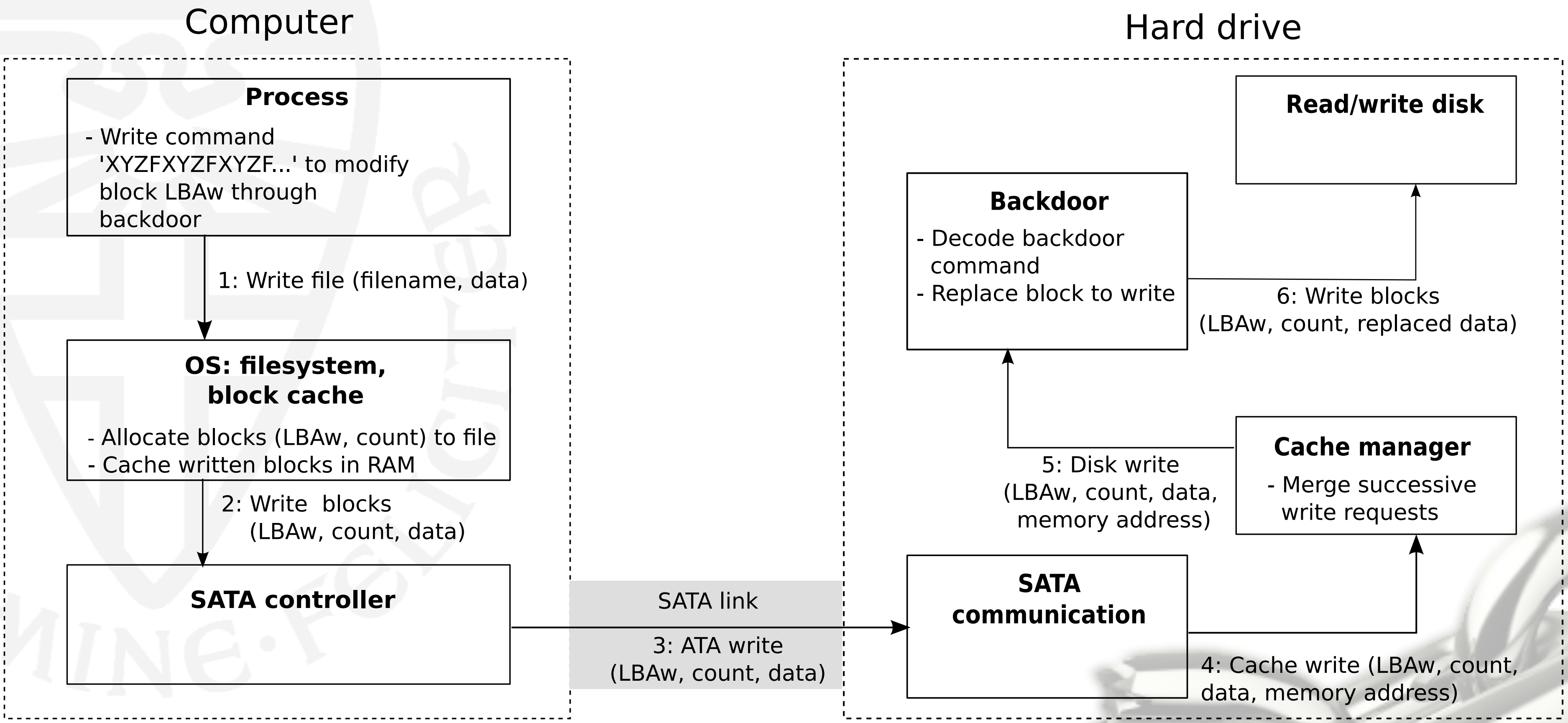


Virtual-machine based rootkits (VMBR)

- SubVirt / Blue Pill
  - Trap a running OS in a thin hypervisor.



Stealth Hard-Drive Backdoor


Computer
Hard drive
Process
- Write command 'XYZFXYZFXYZF...' to modify block LBAw through backdoor
1: Write file (filename, data)
OS: filesystem, block cache
- Allocate blocks (LBAw, count) to file
- Cache written blocks in RAM
2: Write blocks (LBAw, count, data)
SATA controller
SATA link
3: ATA write (LBAw, count, data)
SATA communication
4: Cache write (LBAw, count, data, memory address)
Cache manager
- Merge successive write requests
5: Disk write (LBAw, count, data, memory address)
Backdoor
- Decode backdoor command
- Replace block to write
6: Write blocks (LBAw, count, replaced data)
Read/write disk

Intel manageability engine

- Out-of-band (OOB) communication
  - “Platforms equipped with Intel AMT can be managed remotely, regardless of whether they are powered up and regardless of whether or not they have a functioning operating system.”

Reference:
"Intel Active Management Technology (Intel AMT) Start Here Guide" (PDF).

Reverse-engineering:
http://www.slideshare.net/codeblue_jp/igor-skochinsky-enpub

JIT Spraying

Example: http://en.wikipedia.org/wiki/JIT_spraying

```
var a = (0x11223344^0x44332211^0x44332211^ ...);
```

JIT then will transform bytecode to native x86 code like:

```
0:     b8 44 33 22 11      mov eax,0x11223344
5:     35 11 22 33 44      xor eax,0x44332211
A:     35 11 22 33 44      xor eax,0x44332211
```

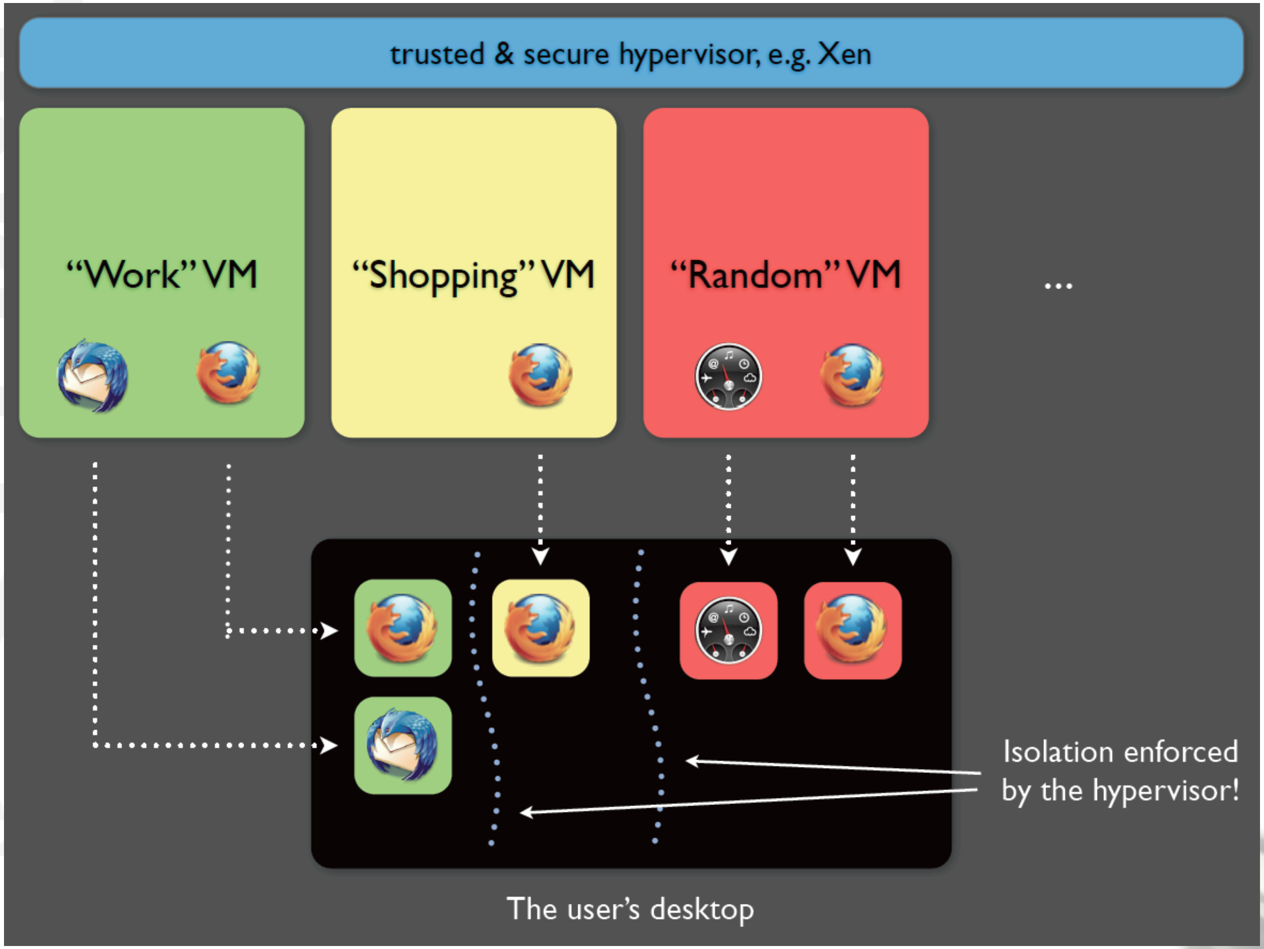
Jumping to the second byte of the "mov" instruction:

```
1:     44                  inc esp
2:     33 22               xor esp,DWORD PTR [edx]
4:     11 35 11 22 33 44   adc DWORD PTR ds:0x44332211,esi
A:     35 11 22 33 44      xor eax,0x44332211
```

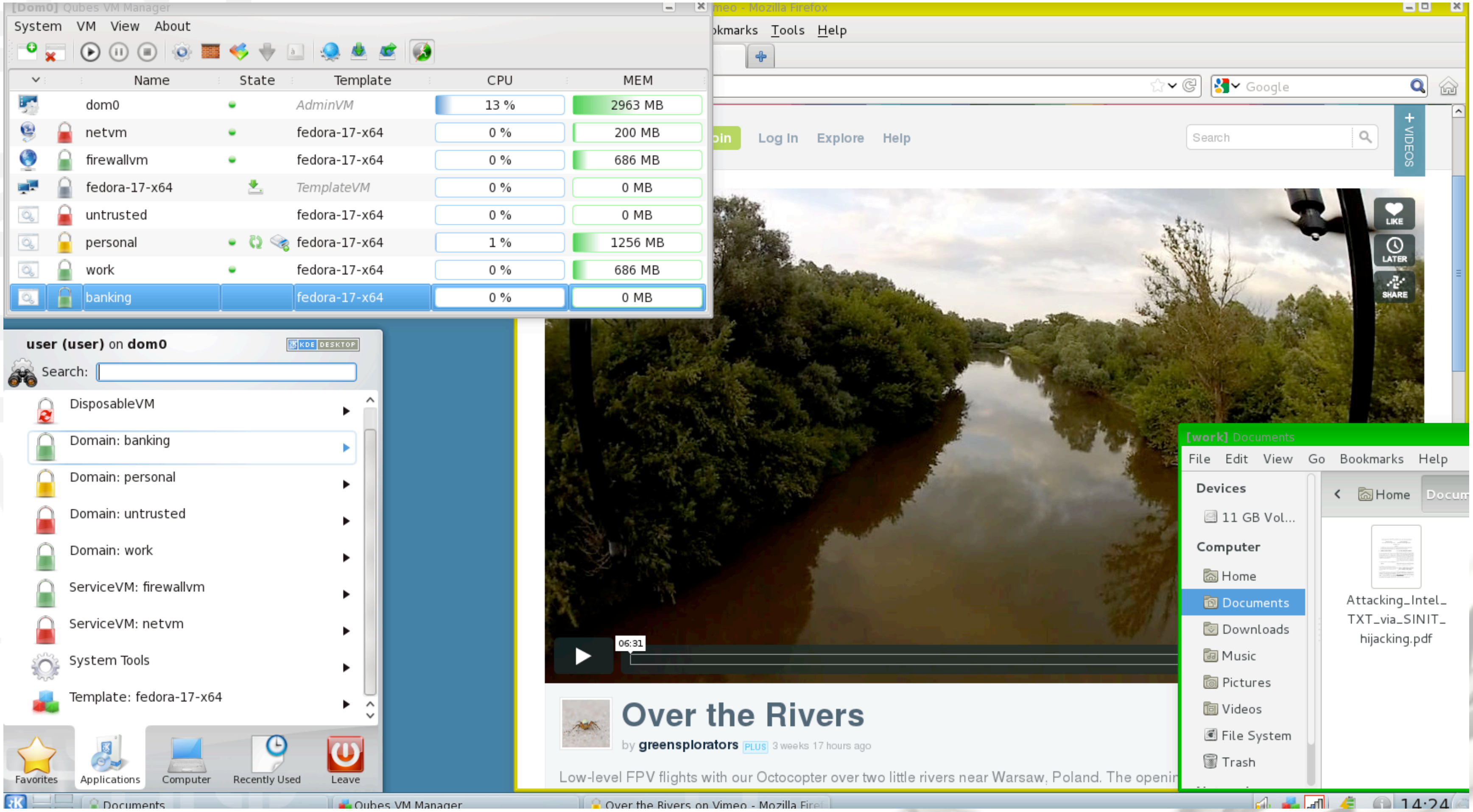
VM Security management

- Re-initialization of “cloned” host
  - Hardware identification (MAC addresses)
  - SSH Private keys
  - Webserver certificates

Security oriented VM (Qubes OS)


trusted & secure hypervisor, e.g. Xen
"Work" VM
"Shopping" VM
"Random" VM
...
Isolation enforced by the hypervisor!
The user's desktop

Security oriented VM (Qubes OS)



[Dom0] Qubes VM Manager

System VM View About

Name	State	Template	CPU	MEM
dom0	●	AdminVM	13 %	2963 MB
netvm	●	fedora-17-x64	0 %	200 MB
firewallvm	●	fedora-17-x64	0 %	686 MB
fedora-17-x64		TemplateVM	0 %	0 MB
untrusted		fedora-17-x64	0 %	0 MB
personal	●	fedora-17-x64	1 %	1256 MB
work	●	fedora-17-x64	0 %	686 MB
banking		fedora-17-x64	0 %	0 MB

user (user) on dom0

Search:

- DisposableVM
- Domain: banking
- Domain: personal
- Domain: untrusted
- Domain: work
- ServiceVM: firewallvm
- ServiceVM: netvm
- System Tools
- Template: fedora-17-x64

Over the Rivers

by greensplorators PLUS 3 weeks 17 hours ago

Low-level FPV flights with our Octocopter over two little rivers near Warsaw, Poland. The openin

[work] Documents

Attacking_Intel_TXT_via_SINIT_hijacking.pdf