IDS for Services Deployed on Virtual Appliances

Read Sprabery

Problem Statement

- Cloud Services are deployed on Virtual Machines.
  - Can we use this fact to provide a defense in depth Intrusion Detection System for services deployed on cloud infrastructure
- Hypervisors can observe
  - Program execution
  - System Calls in the guest kernel
  - Files being read and written to
- What kind of policies can we build around such information?
  - What overhead is caused by collecting the information?
- Can we use VM classifications to group policies?
  - One for dynamic execution engine (PHP, python, cgi)
  - Another for Apache Web Server
  - Database Server

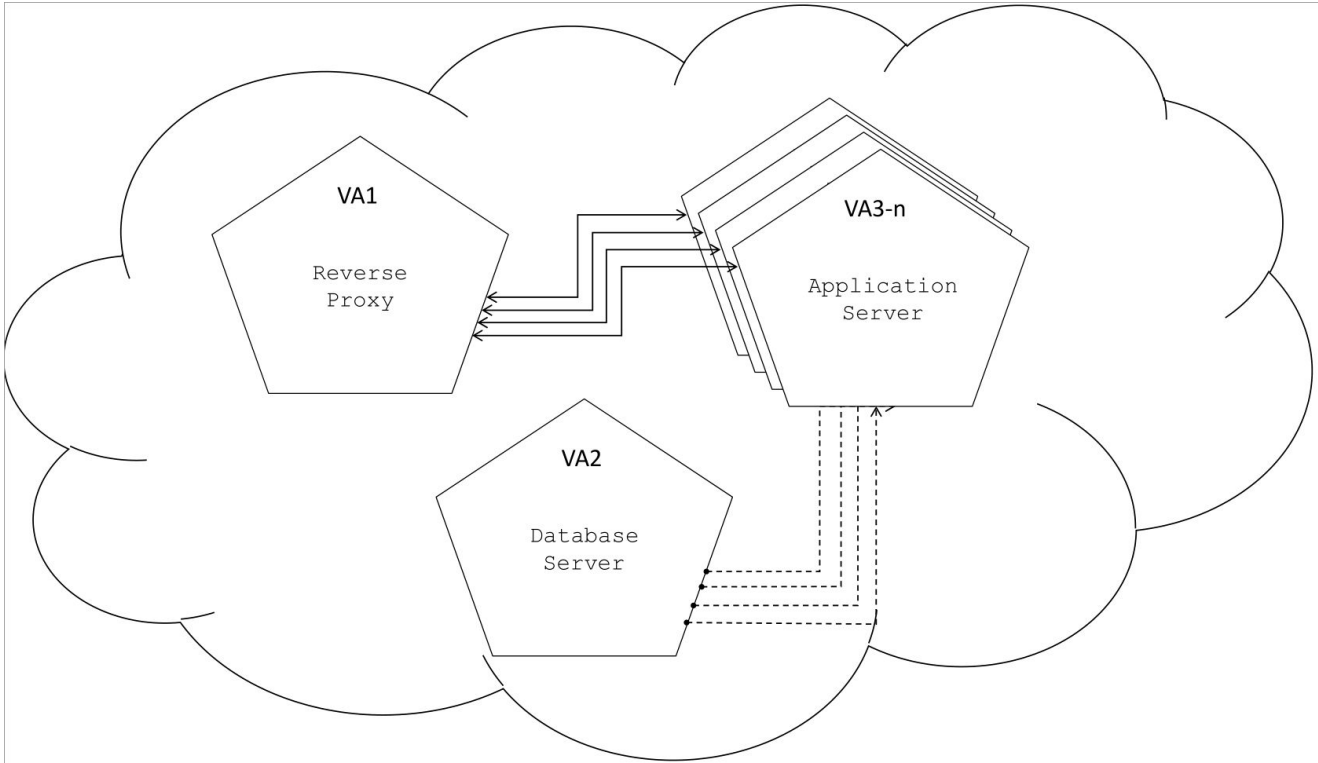
Goals

- Produce a set of probes to find and report relevant information in a manner that can be used by a policy layer
- Develop a policy layer that lets users classify VMs into distinct categories and enforce a set of binaries to be executed and config files that should not be modified
- Develop a representative set of policies for a popular web application such as Wordpress (deployed across 3 virtual machines)
- **Ensure completeness of trusted event log (NEW)**
- Test policies for
  - Performance / Overhead
  - Ease of Use

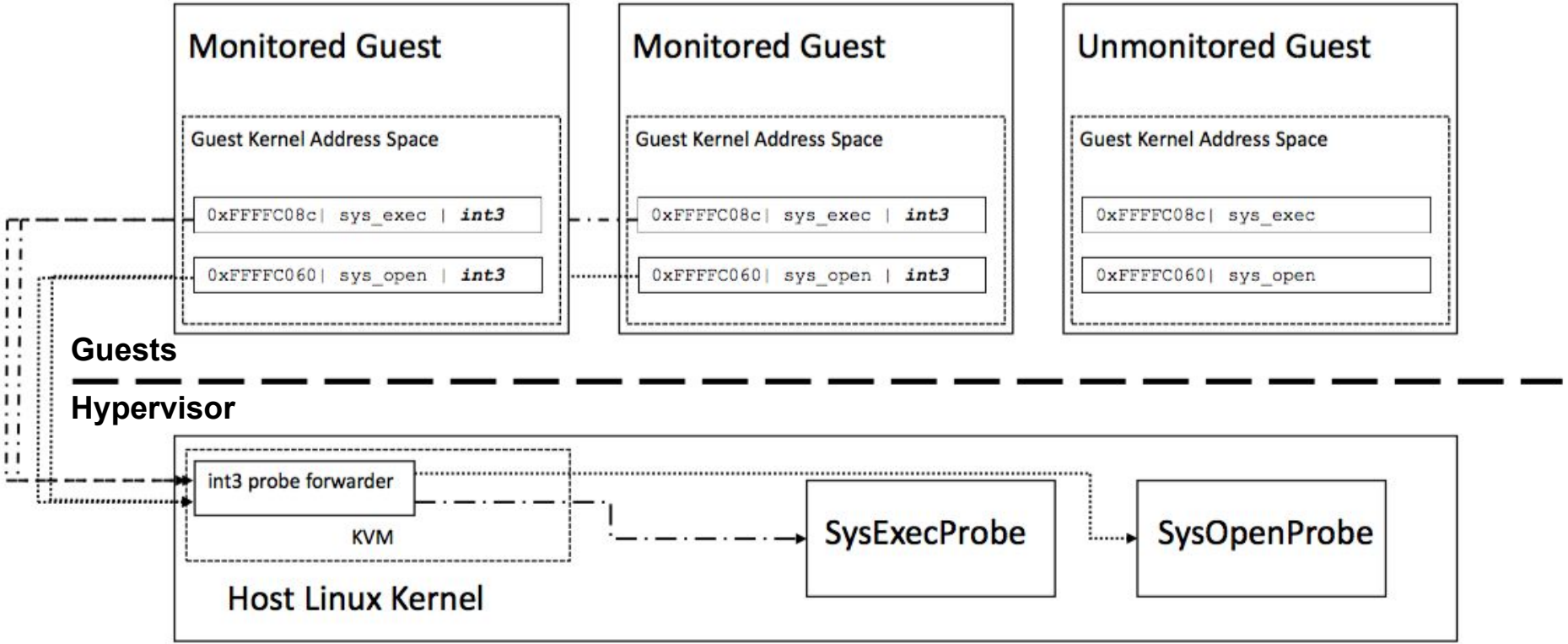
Cloud Deployment of a Typical Web Application

Virtual Appliances (VA) deployed in the cloud

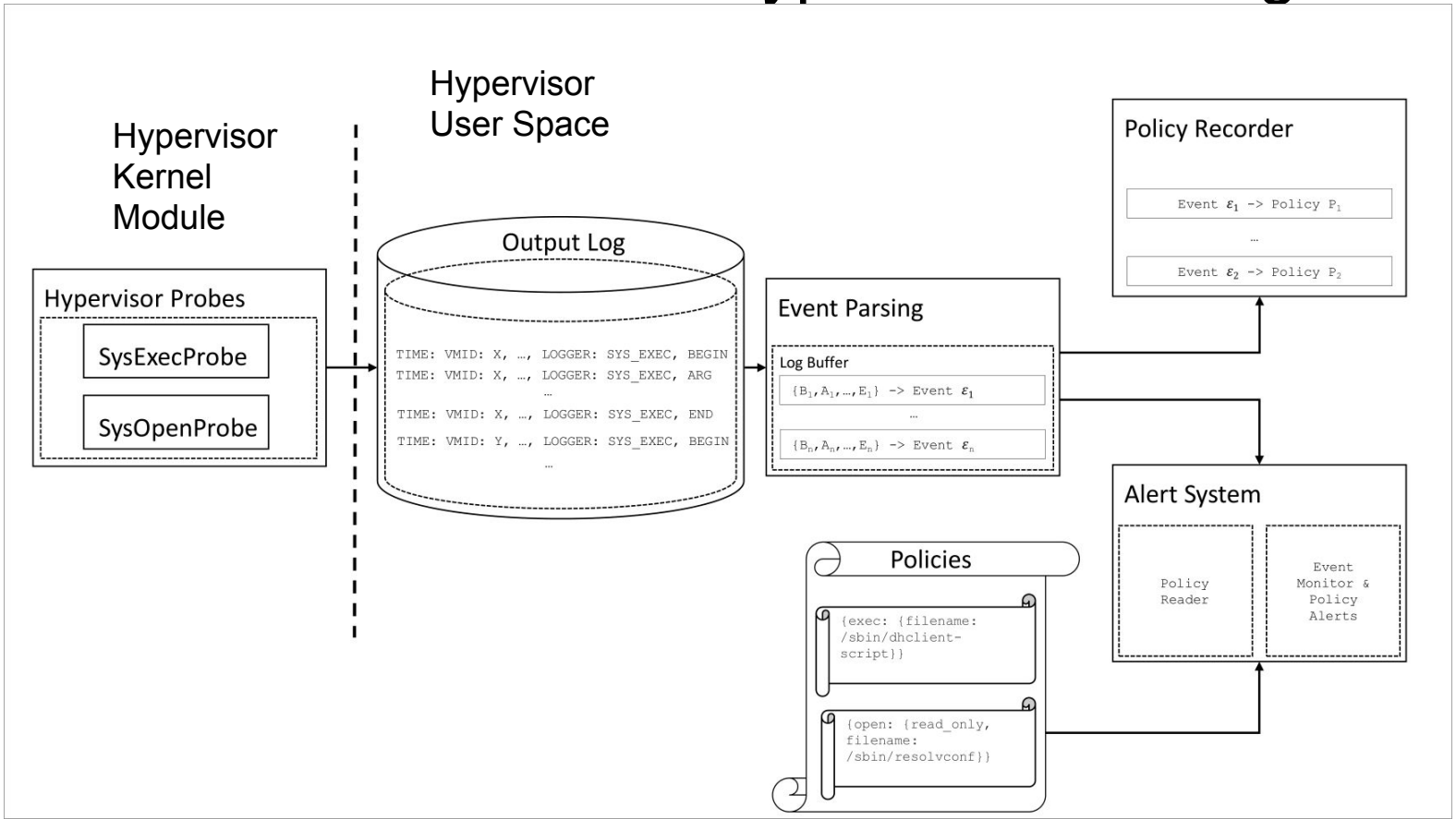
- Solid lines HTTP traffic
- Dashed lines mysql



Probing Overview



Overview of Event Based Hypervisor Probing

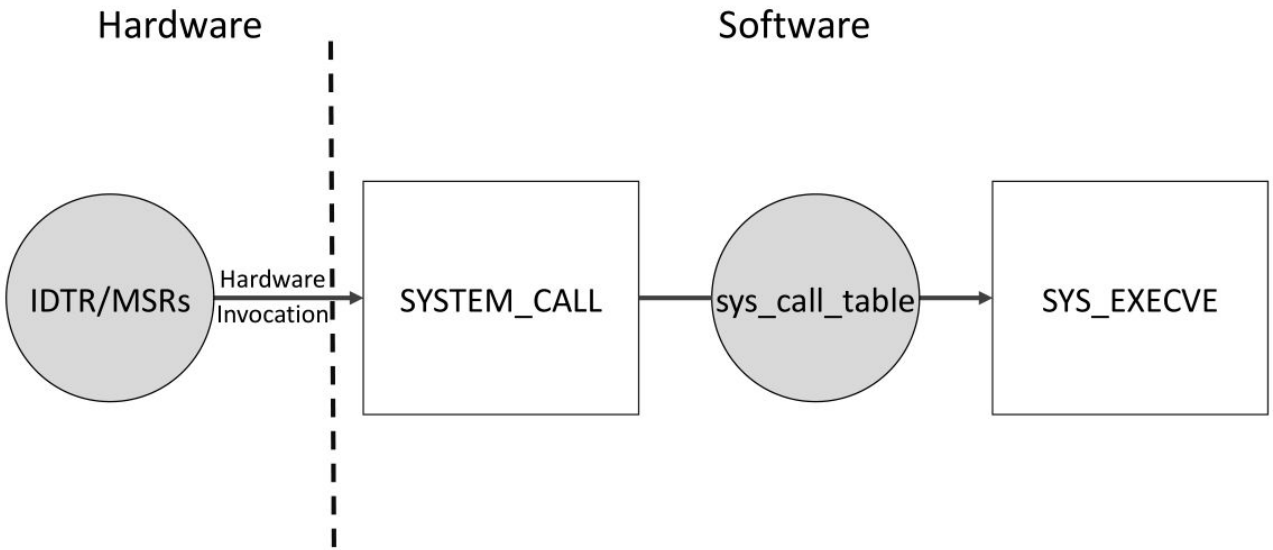


Ensuring the Completeness of Probe Output

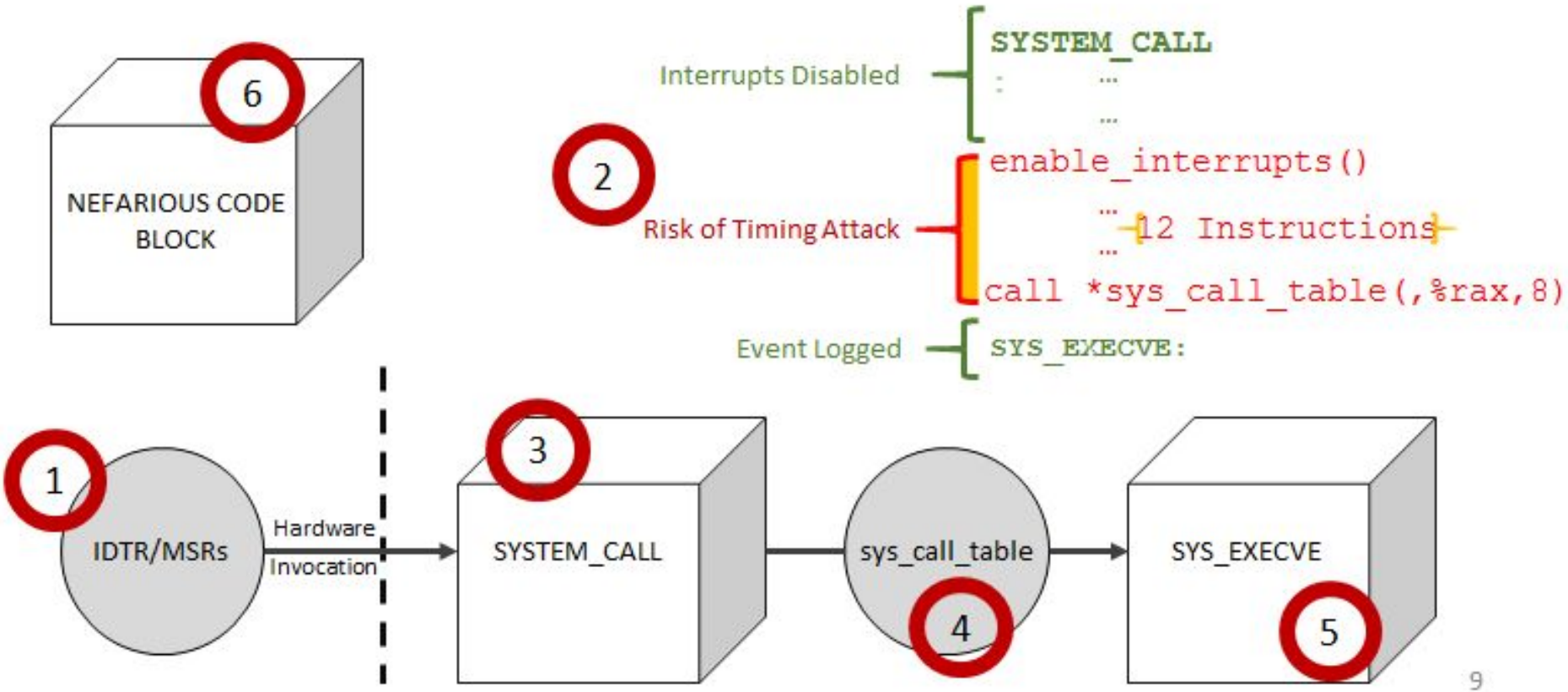
- Completeness - not missing any call to a probed function after the kernel is initialized

- Completeness is essential to ensure policy compliance between the time a guest is booted and the time the probing begins
  - One could quickly inject malware that tampers with the host kernel, and such an event would not appear in the IDS

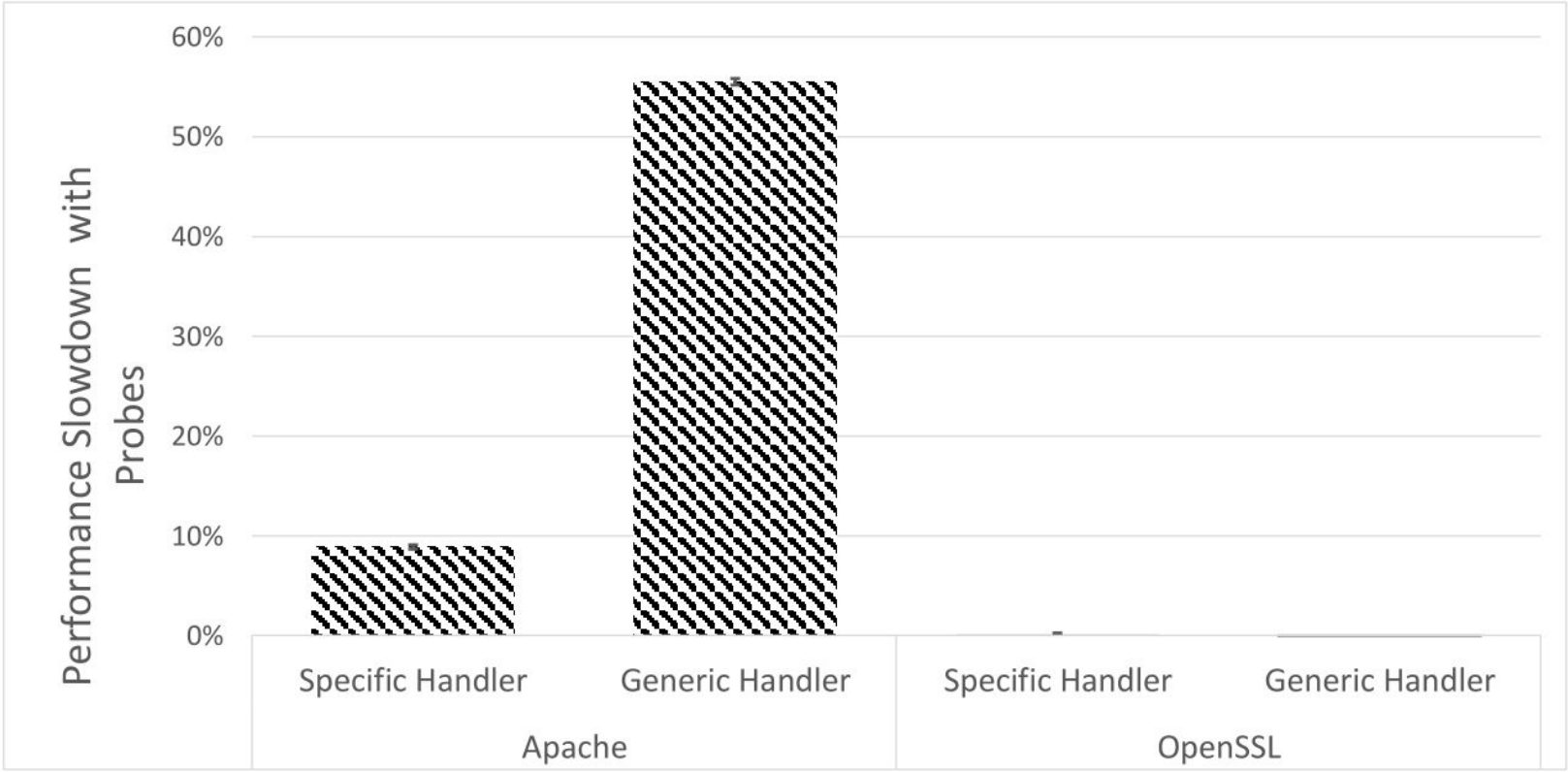
Attack Model



Attack Model



Overhead & Benchmarking

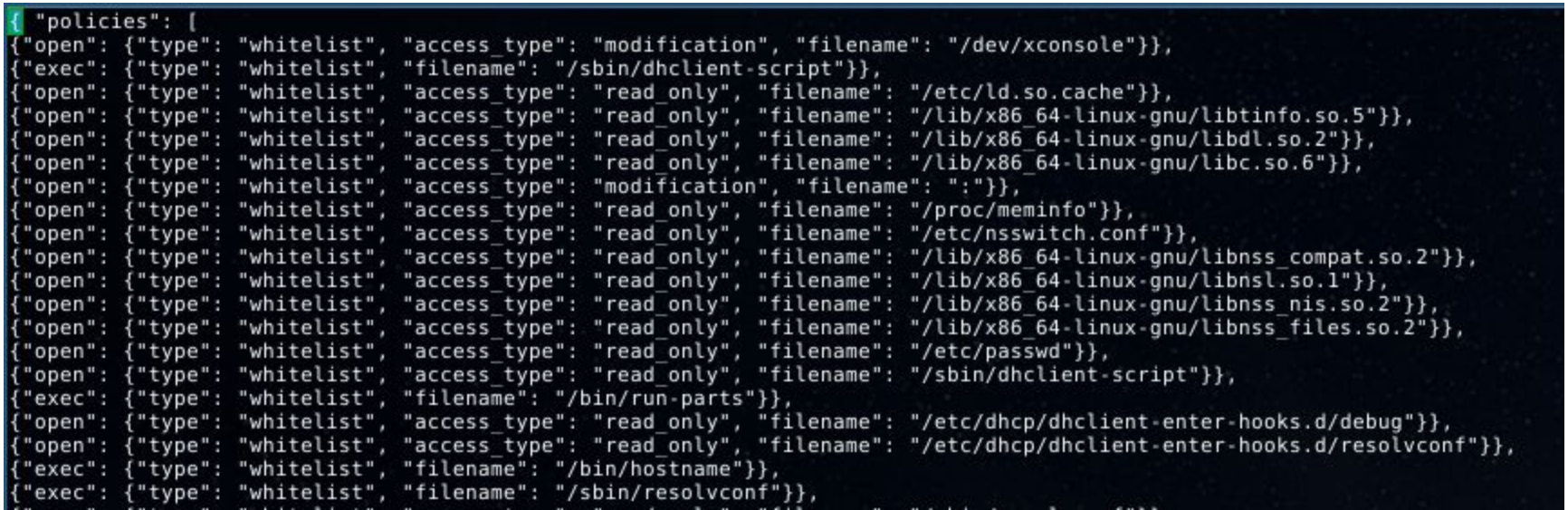


Example Policy

- Requires profiling the guest application
  - Need a representative workload.
- Main portion of Wordpress Policy Below

Example Policy

- Because VA's are built using the same base distro, some policies can be shared
- DHCP Policy Below
  - Runs periodically



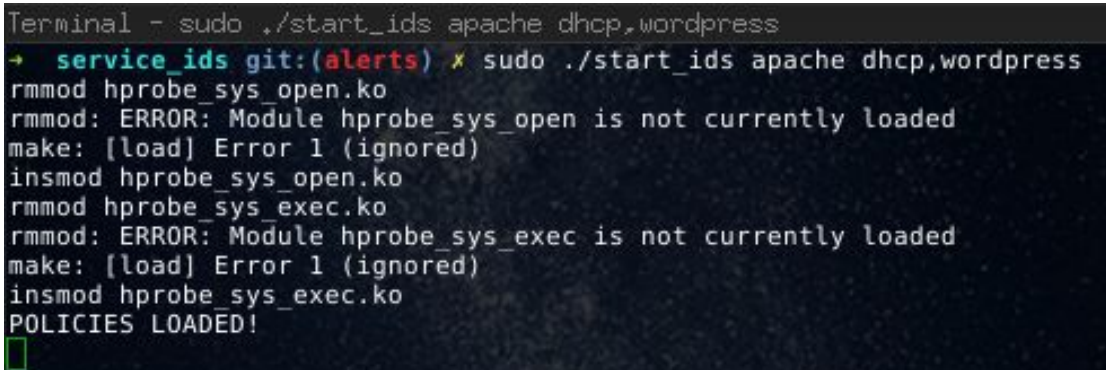
```
{ "policies": [
{"open": {"type": "whitelist", "access_type": "modification", "filename": "/dev/xconsole"}},
{"exec": {"type": "whitelist", "filename": "/sbin/dhclient-script"}},
{"open": {"type": "whitelist", "access_type": "read_only", "filename": "/etc/ld.so.cache"}},
{"open": {"type": "whitelist", "access_type": "read_only", "filename": "/lib/x86_64-linux-gnu/libtinfo.so.5"}},
{"open": {"type": "whitelist", "access_type": "read_only", "filename": "/lib/x86_64-linux-gnu/libdl.so.2"}},
{"open": {"type": "whitelist", "access_type": "read_only", "filename": "/lib/x86_64-linux-gnu/libc.so.6"}},
{"open": {"type": "whitelist", "access_type": "modification", "filename": ":"}},
{"open": {"type": "whitelist", "access_type": "read_only", "filename": "/proc/meminfo"}},
{"open": {"type": "whitelist", "access_type": "read_only", "filename": "/etc/nsswitch.conf"}},
{"open": {"type": "whitelist", "access_type": "read_only", "filename": "/lib/x86_64-linux-gnu/libnss_compat.so.2"}},
{"open": {"type": "whitelist", "access_type": "read_only", "filename": "/lib/x86_64-linux-gnu/libnsl.so.1"}},
{"open": {"type": "whitelist", "access_type": "read_only", "filename": "/lib/x86_64-linux-gnu/libnss_nis.so.2"}},
{"open": {"type": "whitelist", "access_type": "read_only", "filename": "/lib/x86_64-linux-gnu/libnss_files.so.2"}},
{"open": {"type": "whitelist", "access_type": "read_only", "filename": "/etc/passwd"}},
{"open": {"type": "whitelist", "access_type": "read_only", "filename": "/sbin/dhclient-script"}},
{"exec": {"type": "whitelist", "filename": "/bin/run-parts"}},
{"open": {"type": "whitelist", "access_type": "read_only", "filename": "/etc/dhcp/dhclient-enter-hooks.d/debug"}},
{"open": {"type": "whitelist", "access_type": "read_only", "filename": "/etc/dhcp/dhclient-enter-hooks.d/resolvconf"}},
{"exec": {"type": "whitelist", "filename": "/bin/hostname"}},
{"exec": {"type": "whitelist", "filename": "/sbin/resolvconf"}},
```

Attack Scenario & Results

- Wordpress vulnerability
  - AJAX Load More Plugin
    - Allowed a user to inject arbitrary php code.
    - User passwords for wordpress accounts easy to guess
    - Full control over the www-data user (web server user)
- Linux Mint - ISO’s compromised through wordpress vulnerability
  - Attackers used a similar wordpress vulnerability to get shell as the www-data user
  - URL’s for ISO’s were then changed to a malicious image

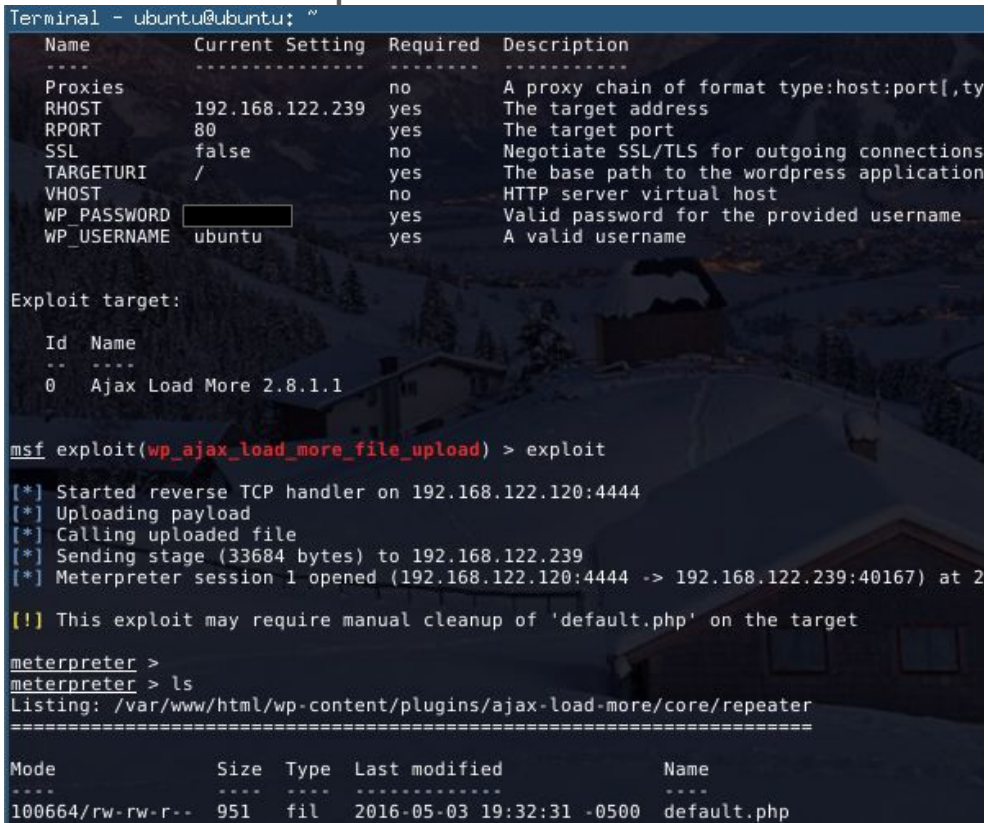
Attack Scenario & Results

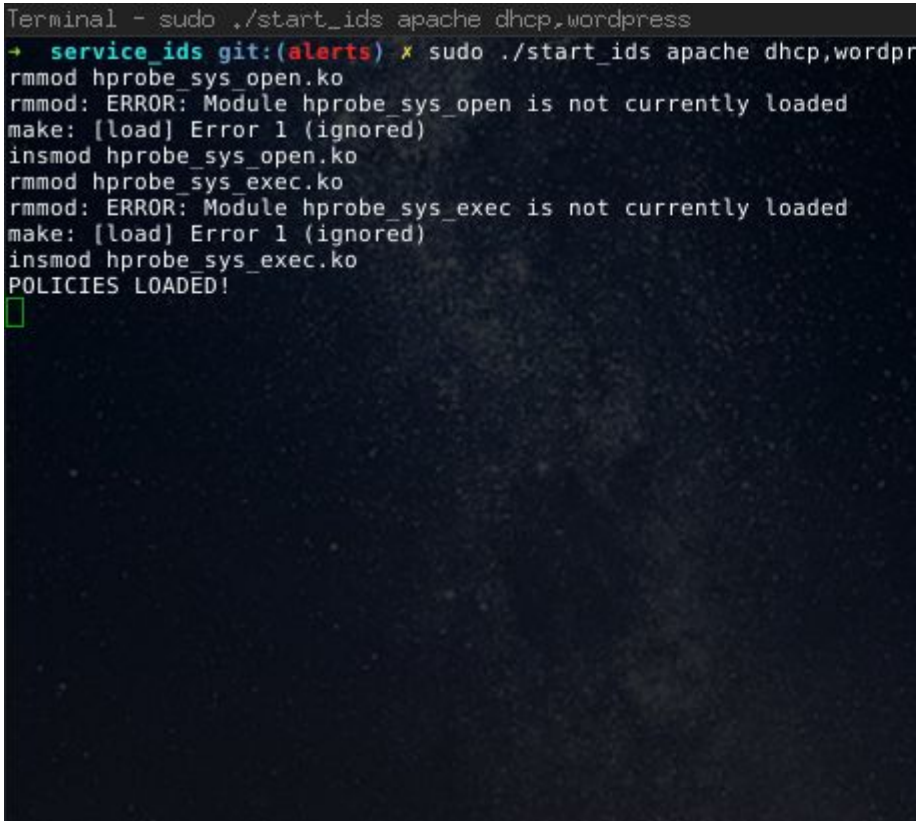
- Start IDS



Attack Scenario & Results

- Run Exploit





Attack Scenario & Results

- Attacker begins investigating the victim

```
meterpreter > ps

Process List
============

 PID   Name                              Arch  User      Path
 ---   ----                              ----  ----      ----
 1     /sbin/init                              root      /sbin/init
 2     [kthreadd]                              root      [kthreadd]
 3     [ksoftirqd/0]                           root      [ksoftirqd/0]
 5     [kworker/0:0H]                          root      [kworker/0:0H]
 6     [kworker/u2:0]                          root      [kworker/u2:0]
 7     [rcu_sched]                             root      [rcu_sched]
 8     [rcuos/0]                               root      [rcuos/0]
 9     [rcu_bh]                                root      [rcu_bh]
 10    [rcuob/0]                               root      [rcuob/0]
 11    [migration/0]                           root      [migration/0]
 12    [watchdog/0]                            root      [watchdog/0]
 13    [khelper]                               root      [khelper]
 14    [kdevtmpfs]                             root      [kdevtmpfs]
 15    [netns]                                 root      [netns]
 16    [writeback]                             root      [writeback]
 17    [kintegrityd]                           root      [kintegrityd]
 18    [bioset]                                root      [bioset]
 19    [kworker/u3:0]                          root      [kworker/u3:0]
 20    [kblockd]                               root      [kblockd]
 21    [ata_sff]                               root      [ata_sff]
 22    [khubd]                                 root      [khubd]
 23    [md]                                    root      [md]
 24    [devfreq_wq]                            root      [devfreq_wq]
 25    [kworker/0:1]                           root      [kworker/0:1]
 26    [khungtaskd]                            root      [khungtaskd]
```

```
Terminal - sudo ./start_ids apache dhcp,wordpress
ALERT! The following event violated a policy!
{"open": {"type": "whitelist", "access_type": "read_only", "filename": "/proc/1030/cmd
ALERT! The following event violated a policy!
{"open": {"type": "whitelist", "access_type": "read_only", "filename": "/proc/1034/sta
ALERT! The following event violated a policy!
{"open": {"type": "whitelist", "access_type": "read_only", "filename": "/proc/1034/sta
ALERT! The following event violated a policy!
{"open": {"type": "whitelist", "access_type": "read_only", "filename": "/proc/1034/cmd
ALERT! The following event violated a policy!
{"open": {"type": "whitelist", "access_type": "read_only", "filename": "/proc/1035/sta
ALERT! The following event violated a policy!
{"open": {"type": "whitelist", "access_type": "read_only", "filename": "/proc/1035/sta
ALERT! The following event violated a policy!
{"open": {"type": "whitelist", "access_type": "read_only", "filename": "/proc/1035/cmd
ALERT! The following event violated a policy!
{"open": {"type": "whitelist", "access_type": "read_only", "filename": "/proc/1036/sta
ALERT! The following event violated a policy!
{"open": {"type": "whitelist", "access_type": "read_only", "filename": "/proc/1036/sta
ALERT! The following event violated a policy!
{"open": {"type": "whitelist", "access_type": "read_only", "filename": "/proc/1036/cmd
ALERT! The following event violated a policy!
{"open": {"type": "whitelist", "access_type": "read_only", "filename": "/proc/1037/sta
ALERT! The following event violated a policy!
{"open": {"type": "whitelist", "access_type": "read_only", "filename": "/proc/1037/sta
ALERT! The following event violated a policy!
{"open": {"type": "whitelist", "access_type": "read_only", "filename": "/proc/1037/cmd
ALERT! The following event violated a policy!
{"open": {"type": "whitelist", "access_type": "read_only", "filename": "/proc/1104/sta
ALERT! The following event violated a policy!
{"open": {"type": "whitelist", "access_type": "read_only", "filename": "/proc/1104/sta
ALERT! The following event violated a policy!
{"open": {"type": "whitelist", "access_type": "read_only", "filename": "/proc/1104/cmd
ALERT! The following event violated a policy!
{"open": {"type": "whitelist", "access_type": "read_only", "filename": "/proc/1105/sta
ALERT! The following event violated a policy!
{"open": {"type": "whitelist", "access_type": "read_only", "filename": "/proc/1105/sta
ALERT! The following event violated a policy!
```

Attack Scenario & Results

- Attacker opens shell

```
meterpreter > shell
Process 1106 created.
Channel 0 created.
ls
default.php
```

```
ALERT! The following event violated a policy!
{"exec": {"type": "whitelist", "filename": "/bin/sh"}}
ALERT! The following event violated a policy!
{"exec": {"type": "whitelist", "filename": "/bin/ls"}}
ALERT! The following event violated a policy!
{"open": {"type": "whitelist", "access_type": "read_only", "filename": "/lib/x86_64-linux-gnu/libacl.so.1"}}
ALERT! The following event violated a policy!
{"open": {"type": "whitelist", "access_type": "read_only", "filename": "/lib/x86_64-linux-gnu/libattr.so.1"}}
```

Summary

- Low Overhead
- Log Completeness
  - Uses EPT faults to load probes when instructions are loaded
- Policies prove to be effective
  - And useful against real world attacks