Hybrid-Bridge:

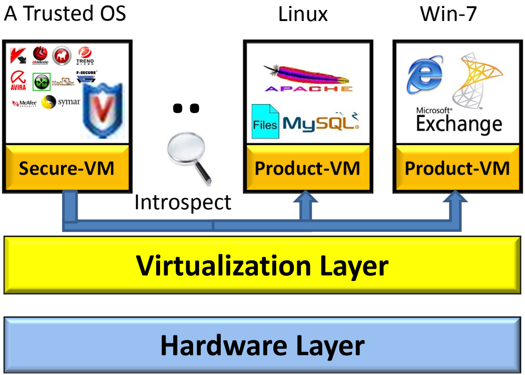
Efficiently Bridging the Semantic-Gap in VMI via Decoupled Execution and Training Memoization

Alireza Saberi, Yangchun Fu, Zhiqiang Lin

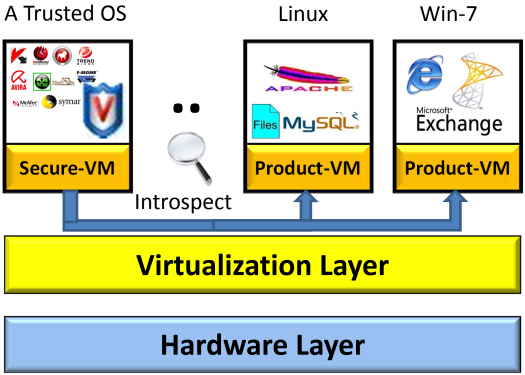
Department of Computer Science
The University of Texas at Dallas

February 24th, 2014

Virtual Machine Introspection (VMI) [Garfinkel et al, NDSS'03]

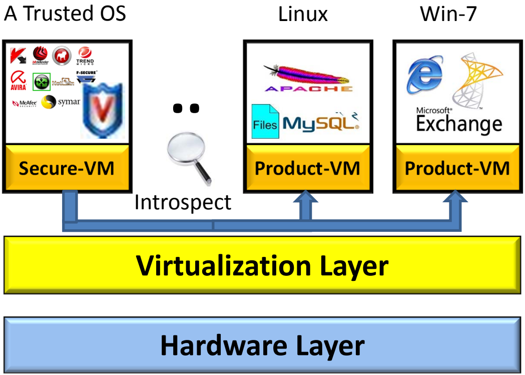


Virtual Machine Introspection (VMI) [Garfinkel et al, NDSS'03]



Using a trusted, dedicated virtualization layer program to monitor the running VMs

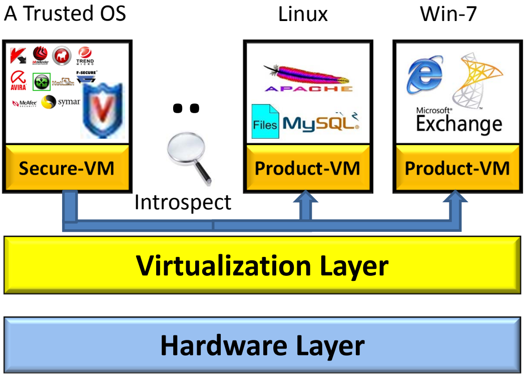
Virtual Machine Introspection (VMI) [Garfinkel et al, NDSS'03]



Using a trusted, dedicated virtualization layer program to monitor the running VMs

- Intrusion Detection
- Malware Analysis
- Memory Forensics

Virtual Machine Introspection (VMI) [Garfinkel et al, NDSS'03]

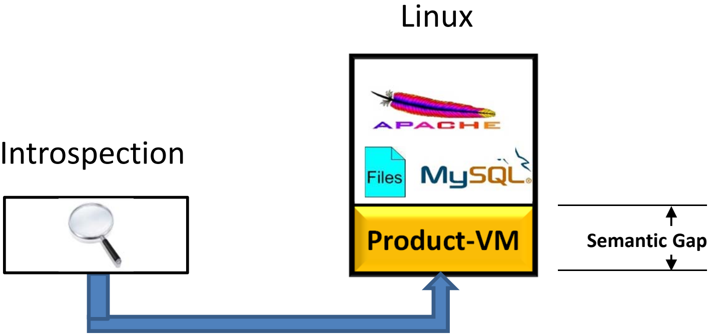


Using a trusted, dedicated virtualization layer program to monitor the running VMs

- Intrusion Detection
- Malware Analysis
- Memory Forensics

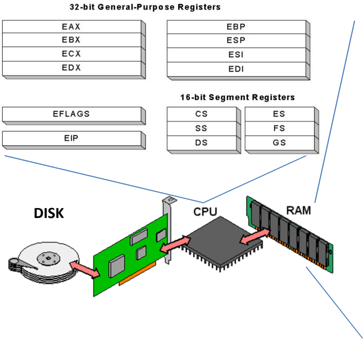
Semantic Gap Problem

The Semantic Gap in VMI ([Chen and Noble HotOS'01])



- View exposed by Virtual Machine Monitor is at low-level
- There is no abstraction and no APIs
- Need to reconstruct the guest-OS abstraction

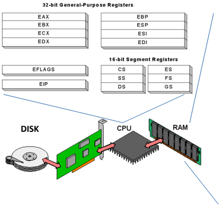
Example: Inspect `pid`s of Guest Memory from VMM



```
…
00001800  eb 40 1b 02 63 74 00 f0  00 00 00 00 00 00 00 00  |.@..ct..........|
00001810  00 00 00 00 80 00 00 00  00 00 00 00 00 00 00 00  |................|
00001820  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
00001830  00 00 00 00 00 00 00 00  10 76 16 cc 00 00 00 00  |.........v......|
00001840  00 19 66 8c d0 50 b8 08  00 00 00 66 8e d0 53 8b  |..f..P.....f..S.|
00001850  d9 ff 2d 19 02 00 00 0f  20 c0 0f ba f0 1f 0f 22  |..-..... ......"|
00001860  c0 eb 00 b9 80 00 00 c0  0f 32 0f ba f0 08 0f 30  |.........2.....0|
00001870  0f 20 e0 0f ba f0 05 0f  22 e0 60 9c 8b d3 c1 ea  |. ......".`.....|
00001880  04 89 a3 76 02 00 00 0f  01 83 80 02 00 00 0f 01  |...v............|
00001890  8b 88 02 00 00 8b 8b 3c  00 00 00 0b c9 74 12 8b  |.......<.....t..|
000018a0  b3 38 00 00 00 8b fb 81  c7 00 30 00 00 2b f9 f3  |.8........0..+..|
000018b0  a4 0f 01 9b 90 02 00 00  0f 01 93 68 02 00 00 66  |...........h...f|
000018c0  b8 10 00 66 8e d8 66 8e  c0 66 8e d0 66 8e e0 66  |...f..f..f..f..f|
…
*
00100f60  00 00 00 00 00 00 00 00  00 f0 ff 5d 76 e3 f0 2f  |...........]v../|
00100f70  93 c9 a4 1d f9 48 be f8  6c c7 1d 92 4c 1e 6e 35  |.....H..l...L.n5|
00100f80  b4 f8 1b ae f6 69 e8 c0  b7 34 74 a1 4e 5a a7 93  |.....i...4t.NZ..|
00100f90  97 2f f3 47 cf d7 10 df  f0 d6 e3 9b f5 cf a9 23  |./.G...........#|
00100fa0  cd 9f 87 4f 37 7f 1e f1  fe dc 7d b9 f9 f3 7b ef  |...O7.....}...{.|
00100fb0  cf 95 bf 94 3f 8d 63 9a  cc 8a 36 5b 56 7b d2 76  |....?.c...6[V{.v|
00100fc0  b6 d9 ad ee 61 f6 90 a4  2c 2b 54 66 37 de 3d a9  |....a...,+Tf7.=.|
00100fd0  b9 d9 67 37 1e 7a b5 ce  ef 0c 58 ee 4d 30 d0 9b  |..g7.z....X.M0..|
00100fe0  c0 6e bc e7 3d f3 e7 d0  9a bf a4 82 1b c7 9c f1  |.n..=...........|
00100ff0  db 66 2b d8 38 cb 2a 91  80 ad 7d 25 d8 0a e5 db  |.f+.8.*...}%....|
```

Virtual Machine Monitor Layer

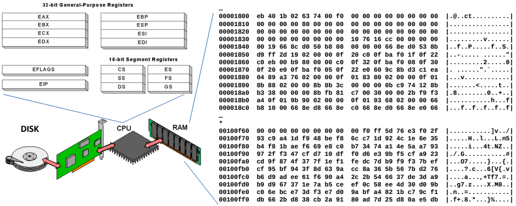
Example: Inspect `pid`s of Guest Memory from VMM



```
…
00001800  eb 40 1b 02 63 74 00 f0  00 00 00 00 00 00 00 00  |.@..ct..........|
00001810  00 00 00 00 80 00 00 00  00 00 00 00 00 00 00 00  |................|
00001820  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
00001830  00 00 00 00 00 00 00 00  10 76 16 cc 00 00 00 00  |.........v......|
00001840  00 19 66 8c d0 50 b8 08  00 00 00 66 8e d0 53 8b  |..f..P.....f..S.|
00001850  d9 ff 2d 19 02 00 00 0f  20 c0 0f ba f0 1f 0f 22  |..-..... ......"|
00001860  c0 eb 00 b9 80 00 00 c0  0f 32 0f ba f0 08 0f 30  |.........2.....0|
00001870  0f 20 e0 0f ba f0 05 0f  22 e0 60 9c 8b d3 c1 ea  |. ......".`.....|
00001880  04 89 a3 76 02 00 00 0f  01 83 80 02 00 00 0f 01  |...v............|
00001890  8b 88 02 00 00 8b 8b 3c  00 00 00 0b c9 74 12 8b  |.......<.....t..|
000018a0  b3 38 00 00 00 8b fb 81  c7 00 30 00 00 2b f9 f3  |.8........0..+..|
000018b0  a4 0f 01 9b 90 02 00 00  0f 01 93 68 02 00 00 66  |...........h...f|
000018c0  b8 10 00 66 8e d8 66 8e  c0 66 8e d0 66 8e e0 66  |...f..f..f..f..f|
…
*
00100f60  00 00 00 00 00 00 00 00  00 f0 ff 5d 76 e3 f0 2f  |...........]v../|
00100f70  93 c9 a4 1d f9 48 be f8  6c c7 1d 92 4c 1e 6e 35  |.....H..l...L.n5|
00100f80  b4 f8 1b ae f6 69 e8 c0  b7 34 74 a1 4e 5a a7 93  |.....i...4t.NZ..|
00100f90  97 2f f3 47 cf d7 10 df  f0 d6 e3 9b f5 cf a9 23  |./.G...........#|
00100fa0  cd 9f 87 4f 37 7f 1e f1  fe dc 7d b9 f9 f3 7b ef  |...O7.....}...{.|
00100fb0  cf 95 bf 94 3f 8d 63 9a  cc 8a 36 5b 56 7b d2 76  |....?.c...6[V{.v|
00100fc0  b6 d9 ad ee 61 f6 90 a4  2c 2b 54 66 37 de 3d a9  |....a...,+Tf7.=.|
00100fd0  b9 d9 67 37 1e 7a b5 ce  ef 0c 58 ee 4d 30 d0 9b  |..g7.z....X.M0..|
00100fe0  c0 6e bc e7 3d f3 e7 d0  9a bf a4 82 1b c7 9c f1  |.n..=...........|
00100ff0  db 66 2b d8 38 cb 2a 91  80 ad 7d 25 d8 0a e5 db  |.f+.8.*...}%....|
```

Virtual Machine Monitor Layer

Example: Inspect `pid`s of Guest Memory from VMM



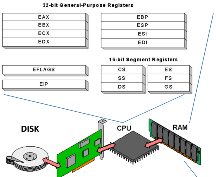
```
…
00001800  eb 40 1b 02 63 74 00 f0  00 00 00 00 00 00 00 00  |.@..ct..........|
00001810  00 00 00 00 80 00 00 00  00 00 00 00 00 00 00 00  |................|
00001820  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
00001830  00 00 00 00 00 00 00 00  10 76 16 cc 00 00 00 00  |.........v......|
00001840  00 19 66 8c d0 50 b8 08  00 00 00 66 8e d0 53 8b  |..f..P.....f..S.|
00001850  d9 ff 2d 19 02 00 00 0f  20 c0 0f ba f0 1f 0f 22  |..-..... ......"|
00001860  c0 eb 00 b9 80 00 00 c0  0f 32 0f ba f0 08 0f 30  |.........2.....0|
00001870  0f 20 e0 0f ba f0 05 0f  22 e0 60 9c 8b d3 c1 ea  |. ......".`.....|
00001880  04 89 a3 76 02 00 00 0f  01 83 80 02 00 00 0f 01  |...v............|
00001890  8b 88 02 00 00 8b 8b 3c  00 00 00 0b c9 74 12 8b  |.......<.....t..|
000018a0  b3 38 00 00 00 8b fb 81  c7 00 30 00 00 2b f9 f3  |.8........0..+..|
000018b0  a4 0f 01 9b 90 02 00 00  0f 01 93 68 02 00 00 66  |...........h...f|
000018c0  b8 10 00 66 8e d8 66 8e  c0 66 8e d0 66 8e e0 66  |...f..f..f..f..f|
…
*
00100f60  00 00 00 00 00 00 00 00  00 f0 ff 5d 76 e3 f0 2f  |...........]v../|
00100f70  93 c9 a4 1d f9 48 be f8  6c c7 1d 92 4c 1e 6e 35  |.....H..l...L.n5|
00100f80  b4 f8 1b ae f6 69 e8 c0  b7 34 74 a1 4e 5a a7 93  |.....i...4t.NZ..|
00100f90  97 2f f3 47 cf d7 10 df  f0 d6 e3 9b f5 cf a9 23  |./.G...........#|
00100fa0  cd 9f 87 4f 37 7f 1e f1  fe dc 7d b9 f9 f3 7b ef  |...O7.....}...{.|
00100fb0  cf 95 bf 94 3f 8d 63 9a  cc 8a 36 5b 56 7b d2 76  |....?.c...6[V{.v|
00100fc0  b6 d9 ad ee 61 f6 90 a4  2c 2b 54 66 37 de 3d a9  |....a...,+Tf7.=.|
00100fd0  b9 d9 67 37 1e 7a b5 ce  ef 0c 58 ee 4d 30 d0 9b  |..g7.z....X.M0..|
00100fe0  c0 6e bc e7 3d f3 e7 d0  9a bf a4 82 1b c7 9c f1  |.n..=...........|
00100ff0  db 66 2b d8 38 cb 2a 91  80 ad 7d 25 d8 0a e5 db  |.f+.8.*...}%....|
```

Virtual Machine Monitor Layer

In Kernel 2.6.18

```
struct task_struct {
      ...
    [188] pid_t pid;
    [192] pid_t tgid;
      ...
    [356] uid_t uid;
    [360] uid_t euid;
    [364] uid_t suid;
    [368] uid_t fsuid;
    [372] gid_t gid;
    [376] gid_t egid;
    [380] gid_t sgid;
    [384] gid_t fsgid;
    ...
    [428] char comm[16];
    ...
}
SIZE: 1408
```

Example: Inspect `pid`s of Guest Memory from VMM



```
…
00001800  eb 40 1b 02 63 74 00 f0  00 00 00 00 00 00 00 00  |.@..ct..........|
00001810  00 00 00 00 80 00 00 00  00 00 00 00 00 00 00 00  |................|
00001820  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
00001830  00 00 00 00 00 00 00 00  10 76 16 cc 00 00 00 00  |.........v......|
00001840  00 19 66 8c d0 50 b8 08  00 00 00 66 8e d0 53 8b  |..f..P.....f..S.|
00001850  d9 ff 2d 19 02 00 00 0f  20 c0 0f ba f0 1f 0f 22  |..-..... ......"|
00001860  c0 eb 00 b9 80 00 00 c0  0f 32 0f ba f0 08 0f 30  |.........2.....0|
00001870  0f 20 e0 0f ba f0 05 0f  22 e0 60 9c 8b d3 c1 ea  |. ......".`.....|
00001880  04 89 a3 76 02 00 00 0f  01 83 80 02 00 00 0f 01  |...v............|
00001890  8b 88 02 00 00 8b 8b 3c  00 00 00 0b c9 74 12 8b  |.......<.....t..|
000018a0  b3 38 00 00 00 8b fb 81  c7 00 30 00 00 2b f9 f3  |.8........0..+..|
000018b0  a4 0f 01 9b 90 02 00 00  0f 01 93 68 02 00 00 66  |...........h...f|
000018c0  b8 10 00 66 8e d8 66 8e  c0 66 8e d0 66 8e e0 66  |...f..f..f..f..f|
…
*
00100f60  00 00 00 00 00 00 00 00  00 f0 ff 5d 76 e3 f0 2f  |...........]v../|
00100f70  93 c9 a4 1d f9 48 be f8  6c c7 1d 92 4c 1e 6e 35  |.....H..l...L.n5|
00100f80  b4 f8 1b ae f6 69 e8 c0  b7 34 74 a1 4e 5a a7 93  |.....i...4t.NZ..|
00100f90  97 2f f3 47 cf d7 10 df  f0 d6 e3 9b f5 cf a9 23  |./.G...........#|
00100fa0  cd 9f 87 4f 37 7f 1e f1  fe dc 7d b9 f9 f3 7b ef  |...O7.....}...{.|
00100fb0  cf 95 bf 94 3f 8d 63 9a  cc 8a 36 5b 56 7b d2 76  |....?.c...6[V{.v|
00100fc0  b6 d9 ad ee 61 f6 90 a4  2c 2b 54 66 37 de 3d a9  |....a...,+Tf7.=.|
00100fd0  b9 d9 67 37 1e 7a b5 ce  ef 0c 58 ee 4d 30 d0 9b  |..g7.z....X.M0..|
00100fe0  c0 6e bc e7 3d f3 e7 d0  9a bf a4 82 1b c7 9c f1  |.n..=...........|
00100ff0  db 66 2b d8 38 cb 2a 91  80 ad 7d 25 d8 0a e5 db  |.f+.8.*...}%....|
```

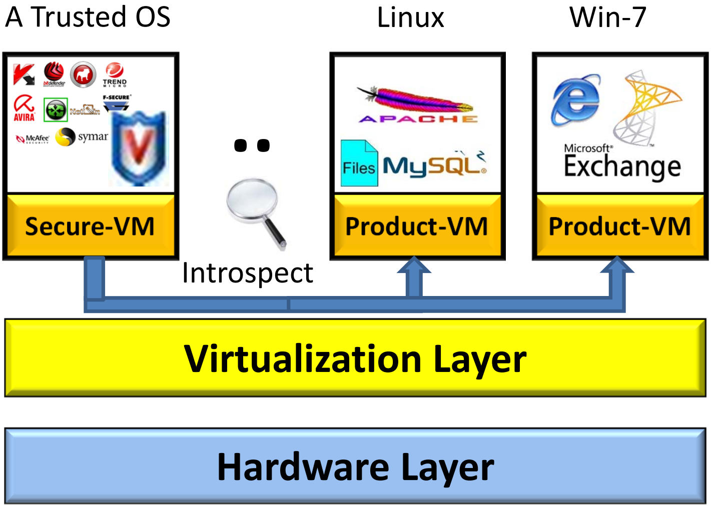
Virtual Machine Monitor Layer

- Kernel specific data structure definition
- Kernel symbols (global variable)
- Virtual to physical (V2P) translation

In Kernel 2.6.18

```
struct task_struct {
    ...
   [188] pid_t pid;
   [192] pid_t tgid;
    ...
   [356] uid_t uid;
   [360] uid_t euid;
   [364] uid_t suid;
   [368] uid_t fsuid;
   [372] gid_t gid;
   [376] gid_t egid;
   [380] gid_t sgid;
   [384] gid_t fsgid;
   ...
   [428] char comm[16];
   ...
}
SIZE: 1408
```

VMI: Reuse Existing Inspection Tools?



VMI: Reuse Existing Inspection Tools? (`sys_getpid`)

```
<sys_getpid>:
<task_tgid_vnr>:
1:  c10583e0: push   %ebp
2:  c10583e1: mov    %esp,%ebp
3:  c10583e3: push   %ebx
4:  c10583e4: sub    $0x14,%esp

// Accessing Global Variable: struct task_strut current_task
5:  c10583e7: mov    %fs:0xc17f34cc,%ebx
        c10583ea: R_386_32   current_task
```

(a)

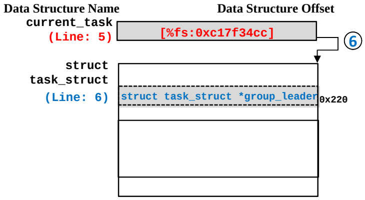
Data Structure Name	Data Structure Offset
current_task (Line: 5)	[%fs:0xc17f34cc]

(b)

VMI: Reuse Existing Inspection Tools? (`sys_getpid`)

```
<sys_getpid>:
<task_tgid_vnr>:
1:  c10583e0: push   %ebp
2:  c10583e1: mov    %esp,%ebp
3:  c10583e3: push   %ebx
4:  c10583e4: sub    $0x14,%esp

// Accessing Global Variable: struct task_strut current_task
5:  c10583e7: mov    %fs:0xc17f34cc,%ebx
        c10583ea: R_386_32   current_task

// Accessing struct task_struct: current_task->group_leader
6:  c10583fe: mov    0x220(%ebx),%eax
```

(a)

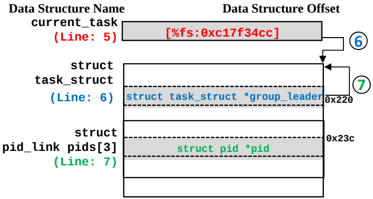


(b)

VMI: Reuse Existing Inspection Tools? (sys_getpid)

```
<sys_getpid>:
<task_tgid_vnr>:
1:  c10583e0: push   %ebp
2:  c10583e1: mov    %esp,%ebp
3:  c10583e3: push   %ebx
4:  c10583e4: sub    $0x14,%esp

// Accessing Global Variable: struct task_strut current_task
5:  c10583e7: mov    %fs:0xc17f34cc,%ebx
        c10583ea: R_386_32   current_task

// Accessing struct task_struct: current_task->group_leader
6:  c10583fe: mov    0x220(%ebx),%eax

// Accessing struct pid: current_task->group_leader->pids[0]->pid
7:  c1058404: mov    0x23c(%eax),%eax
```

(a)

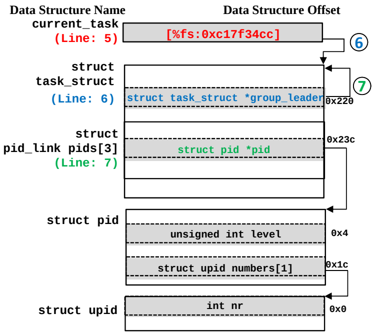


(b)

VMI: Reuse Existing Inspection Tools? (sys_getpid)

```
<sys_getpid>:
<task_tgid_vnr>:
1:  c10583e0: push   %ebp
2:  c10583e1: mov    %esp,%ebp
3:  c10583e3: push   %ebx
4:  c10583e4: sub    $0x14,%esp

// Accessing Global Variable: struct task_strut current_task
5:  c10583e7: mov    %fs:0xc17f34cc,%ebx
        c10583ea: R_386_32    current_task

// Accessing struct task_struct: current_task->group_leader
6:  c10583fe: mov    0x220(%ebx),%eax

// Accessing struct pid: current_task->group_leader->pids[0]->pid
7:  c1058404: mov    0x23c(%eax),%eax

8:  c105840a: call   c1065660 <pid_vnr>
9:  c105840f: add    $0x14,%esp
```

(a)



Data Structure Name

Data Structure Offset

current_task (Line: 5) — [%fs:0xc17f34cc]

struct task_struct (Line: 6) — struct task_struct *group_leader — 0x220

struct pid_link pids[3] (Line: 7) — struct pid *pid — 0x23c

struct pid — unsigned int level — 0x4

struct upid numbers[1] — 0x1c

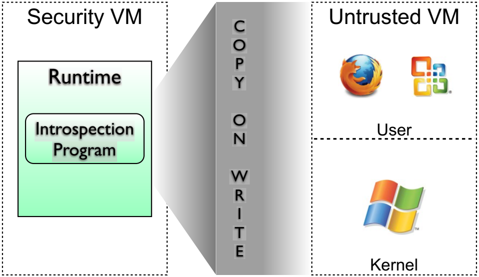
struct upid — int nr — 0x0

(b)

Challenges

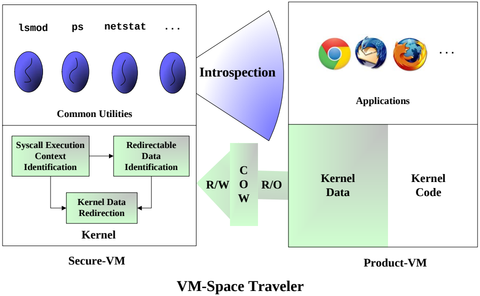
- Redirect Data (Between Secure-VM and Product-VM)
- Find Redirectable Instructions

Virtuoso [Dolan-Gavitt et al, Oakland'11]



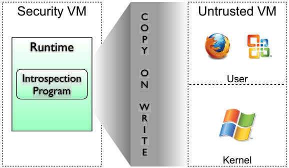
- Train the execution of inspection software
- Suffer from coverage (incompleteness)
- High overhead (140X slowdown)

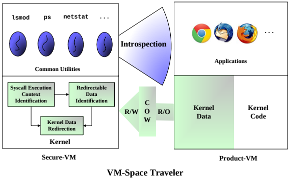
VMST [Fu and Lin, Oakland'12]



- Online kernel data redirection
- Data dependence tracking
- Complete, but w/ very high overhead (hundreds of times of slowdown)

Insight: can we combine Virtuoso and VMST?





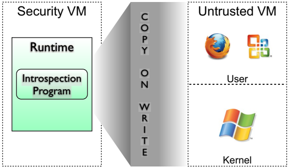
Virutoso

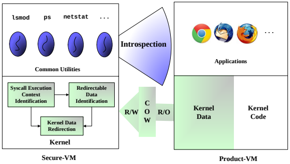
- Training, offline
- Binary code translation

VMST

- Data redirection, online
- Taint analysis

Insight: can we combine Virtuoso and VMST?





VM-Space Traveler

Virutoso

- Training, offline
- Binary code translation

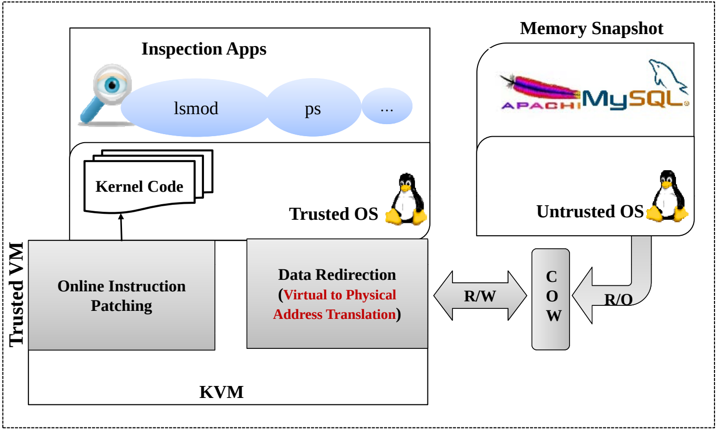
VMST

- Data redirection, online
- Taint analysis

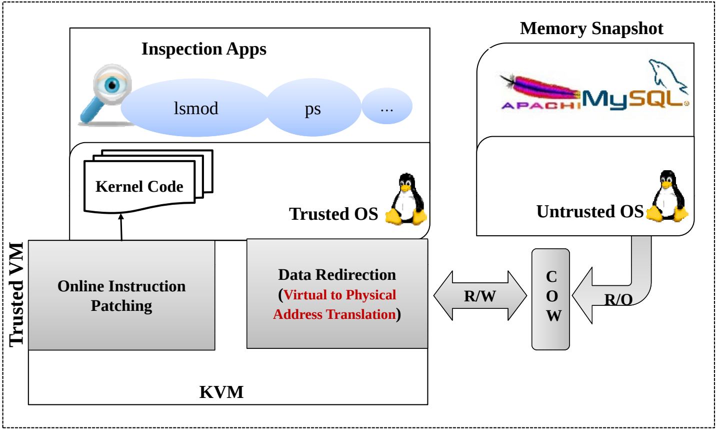
Hybrid

- Decouple the taint analysis
- Combine online and offline with a fallback (much like an OS page fault mechanism) and memoization

Fast-Bridge


Memory Snapshot
Inspection Apps
lsmod
ps
...
Kernel Code
Trusted OS
Untrusted OS
Trusted VM
Online Instruction Patching
Data Redirection (Virtual to Physical Address Translation)
R/W
C O W
R/O
KVM

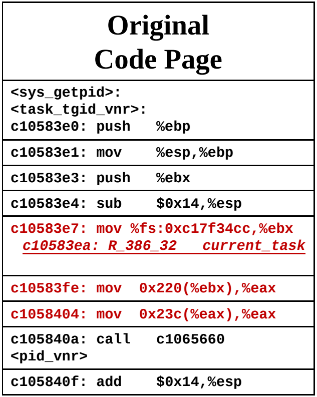
FAST-BRIDGE



Kernel Data Redirection

- Static Kernel Binary Rewriting (hard)
- Dynamic Kernel Binary Instrumentation (slow)

Instruction Patching



Original Code Page

```
<sys_getpid>:
<task_tgid_vnr>:
c10583e0: push   %ebp
c10583e1: mov    %esp,%ebp
c10583e3: push   %ebx
c10583e4: sub    $0x14,%esp
c10583e7: mov %fs:0xc17f34cc,%ebx
  c10583ea: R_386_32    current_task
c10583fe: mov  0x220(%ebx),%eax
c1058404: mov  0x23c(%eax),%eax
c105840a: call   c1065660
<pid_vnr>
c105840f: add    $0x14,%esp
```

Instruction Patching

Original Code Page	Non-Redirectable Code Page	Redirectable Code Page
<sys_getpid>: <task_tgid_vnr>: c10583e0: push %ebp	push %ebp	int 3
c10583e1: mov %esp,%ebp	mov %esp,%ebp	mov %esp,%ebp
c10583e3: push %ebx	push %ebx	int 3
c10583e4: sub $0x14,%esp	sub $0x14,%esp	sub $0x14,%esp
c10583e7: mov %fs:0xc17f34cc,%ebx *c10583ea: R_386_32 current_task*	int 3	mov %fs:0xc17f34cc,%ebx *c10583ea: R_386_32 current_task*
c10583fe: mov 0x220(%ebx),%eax	int 3	mov 0x220(%ebx),%eax
c1058404: mov 0x23c(%eax),%eax	int 3	mov 0x23c(%eax),%eax
c105840a: call c1065660 <pid_vnr>	call c1065660 <pid_vnr>	int 3
c105840f: add $0x14,%esp	add $0x14,%esp	add $0x14,%esp

Instruction Patching

Original Code Page	Non-Redirectable Code Page	Redirectable Code Page
<sys_getpid>: <task_tgid_vnr>: c10583e0: push %ebp	push %ebp	int 3
c10583e1: mov %esp,%ebp	mov %esp,%ebp	mov %esp,%ebp
c10583e3: push %ebx	push %ebx	int 3
c10583e4: sub $0x14,%esp	sub $0x14,%esp	sub $0x14,%esp
c10583e7: mov %fs:0xc17f34cc,%ebx *c10583ea: R_386_32 current_task*	int 3	mov %fs:0xc17f34cc,%ebx *c10583ea: R_386_32 current_task*
c10583fe: mov 0x220(%ebx),%eax	int 3	mov 0x220(%ebx),%eax
c1058404: mov 0x23c(%eax),%eax	int 3	mov 0x23c(%eax),%eax
c105840a: call c1065660 <pid_vnr>	call c1065660 <pid_vnr>	int 3
c105840f: add $0x14,%esp	add $0x14,%esp	add $0x14,%esp

Instruction Patching

Original Code Page	Non-Redirectable Code Page	Redirectable Code Page
<sys_getpid>: <task_tgid_vnr>: c10583e0: push %ebp	push %ebp	int 3
c10583e1: mov %esp,%ebp	mov %esp,%ebp	mov %esp,%ebp
c10583e3: push %ebx	push %ebx	int 3
c10583e4: sub $0x14,%esp	sub $0x14,%esp	sub $0x14,%esp
c10583e7: mov %fs:0xc17f34cc,%ebx *c10583ea: R_386_32 current_task*	int 3 VMexit	mov %fs:0xc17f34cc,%ebx *c10583ea: R_386_32 current_task*
c10583fe: mov 0x220(%ebx),%eax	int 3	mov 0x220(%ebx),%eax
c1058404: mov 0x23c(%eax),%eax	int 3	mov 0x23c(%eax),%eax
c105840a: call c1065660 <pid_vnr>	call c1065660 <pid_vnr>	int 3
c105840f: add $0x14,%esp	add $0x14,%esp	add $0x14,%esp

Instruction Patching

Original Code Page	Non-Redirectable Code Page	Redirectable Code Page
<sys_getpid>: <task_tgid_vnr>: c10583e0: push %ebp	push %ebp	int 3
c10583e1: mov %esp,%ebp	mov %esp,%ebp	mov %esp,%ebp
c10583e3: push %ebx	push %ebx	int 3
c10583e4: sub $0x14,%esp	sub $0x14,%esp	sub $0x14,%esp
c10583e7: mov %fs:0xc17f34cc,%ebx *c10583ea: R_386_32 current_task*	int 3 VMexit	mov %fs:0xc17f34cc,%ebx *c10583ea: R_386_32 current_task*
c10583fe: mov 0x220(%ebx),%eax	int 3	mov 0x220(%ebx),%eax
c1058404: mov 0x23c(%eax),%eax	int 3	mov 0x23c(%eax),%eax
c105840a: call c1065660 <pid_vnr>	call c1065660 <pid_vnr>	int 3
c105840f: add $0x14,%esp	add $0x14,%esp	add $0x14,%esp

Instruction Patching

Original Code Page	Non-Redirectable Code Page	Redirectable Code Page
<sys_getpid>: <task_tgid_vnr>: c10583e0: push %ebp	push %ebp	int 3
c10583e1: mov %esp,%ebp	mov %esp,%ebp	mov %esp,%ebp
c10583e3: push %ebx	push %ebx	int 3
c10583e4: sub $0x14,%esp	sub $0x14,%esp	sub $0x14,%esp
c10583e7: mov %fs:0xc17f34cc,%ebx *c10583ea: R_386_32 current_task*	int 3 VMexit	mov %fs:0xc17f34cc,%ebx *c10583ea: R_386_32 current_task*
c10583fe: mov 0x220(%ebx),%eax	int 3	mov 0x220(%ebx),%eax
c1058404: mov 0x23c(%eax),%eax	int 3	mov 0x23c(%eax),%eax
c105840a: call c1065660 <pid_vnr>	call c1065660 <pid_vnr> VMexit	int 3
c105840f: add $0x14,%esp	add $0x14,%esp	add $0x14,%esp

Instruction Patching

Original Code Page	Non-Redirectable Code Page	Redirectable Code Page
<sys_getpid>: <task_tgid_vnr>: c10583e0: push %ebp	push %ebp	int 3
c10583e1: mov %esp,%ebp	mov %esp,%ebp	mov %esp,%ebp
c10583e3: push %ebx	push %ebx	int 3
c10583e4: sub $0x14,%esp	sub $0x14,%esp	sub $0x14,%esp
c10583e7: mov %fs:0xc17f34cc,%ebx *c10583ea: R_386_32 current_task*	int 3 VMexit	mov %fs:0xc17f34cc,%ebx *c10583ea: R_386_32 current_task*
c10583fe: mov 0x220(%ebx),%eax	int 3	mov 0x220(%ebx),%eax
c1058404: mov 0x23c(%eax),%eax	int 3	mov 0x23c(%eax),%eax
c105840a: call c1065660 <pid_vnr>	call c1065660 <pid_vnr> VMexit	int 3
c105840f: add $0x14,%esp	add $0x14,%esp	add $0x14,%esp

HYBRID-BRIDGE: Architecture Overview

Challenges

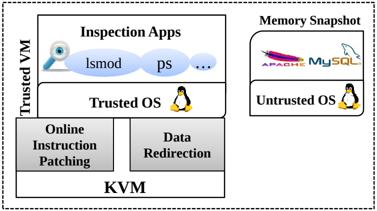
- Redirect Data (Between Secure-VM and Product-VM)
- Find Redirectable Instructions

HYBRID-BRIDGE: Architecture Overview

HYBRID-BRIDGE

HYBRID-BRIDGE: Architecture Overview

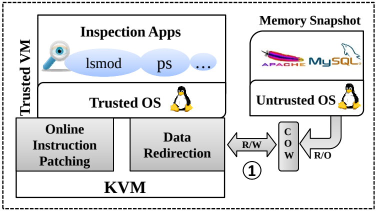
FAST-BRIDGE



HYBRID-BRIDGE

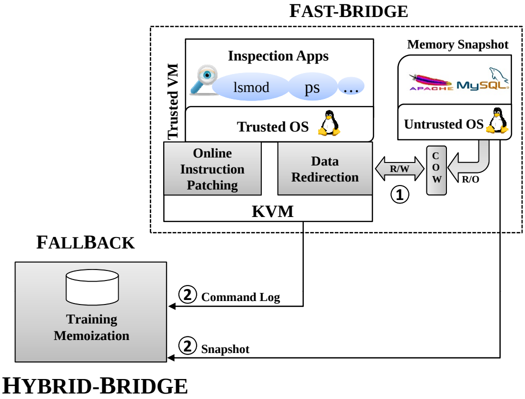
HYBRID-BRIDGE: Architecture Overview

FAST-BRIDGE

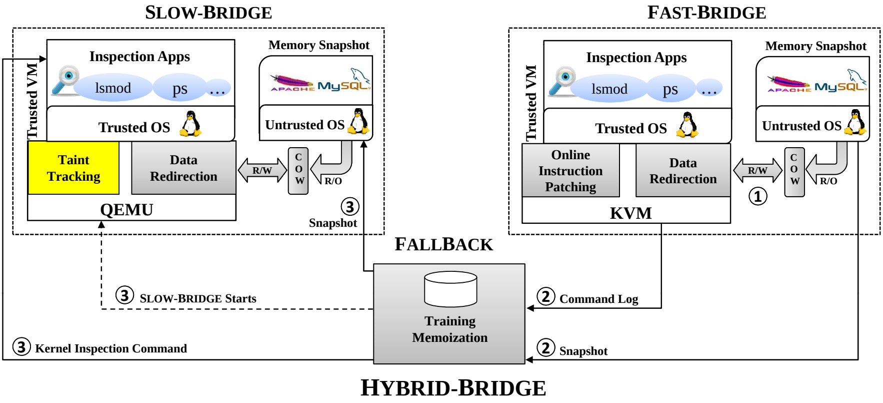


HYBRID-BRIDGE

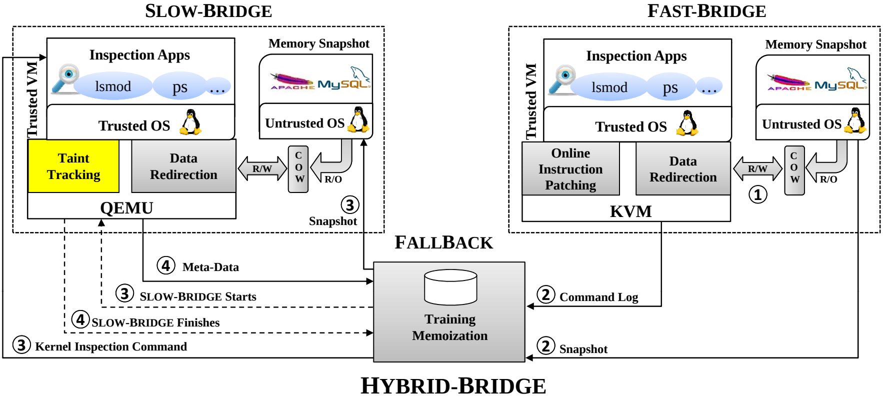
HYBRID-BRIDGE: Architecture Overview



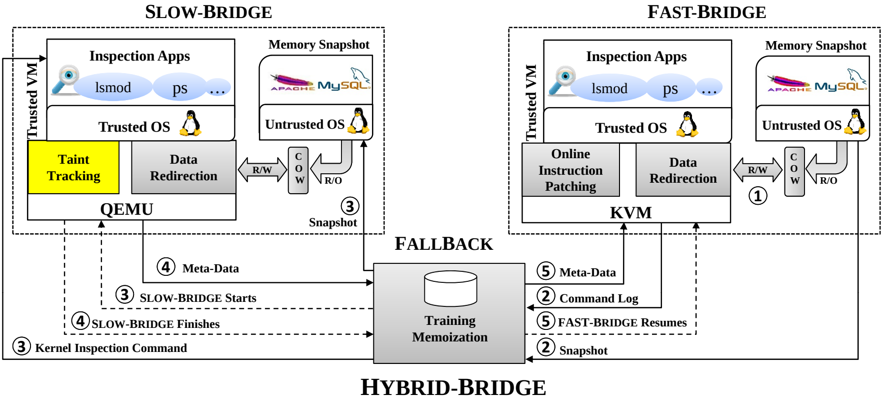
HYBRID-BRIDGE: Architecture Overview



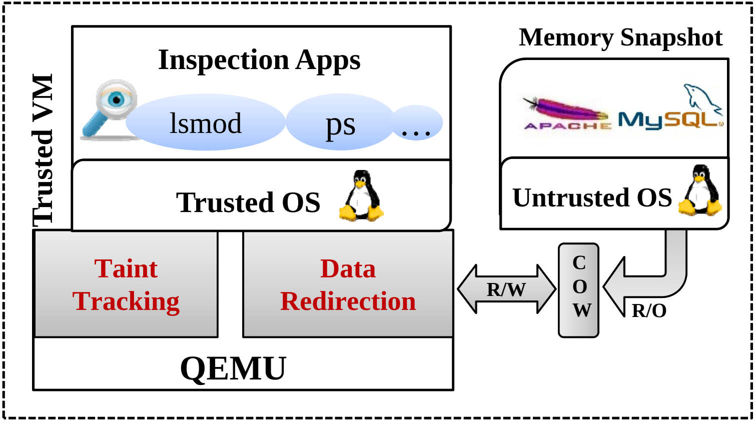
HYBRID-BRIDGE: Architecture Overview



HYBRID-BRIDGE: Architecture Overview



SLOW-BRIDGE


Trusted VM
Inspection Apps
lsmod
ps
...
Trusted OS
Memory Snapshot
APACHE
MySQL
Untrusted OS
Taint Tracking
Data Redirection
R/W
C O W
R/O
QEMU

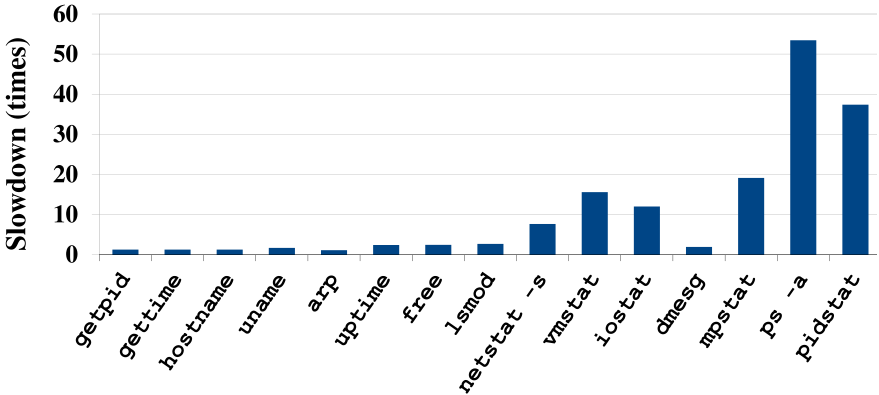
Experiment Setup

- 15 native inspection tools
- VMST, VIRTUOSO, HYBRID-BRIDGE
- Guest: Ubuntu 12.04 (kernel 2.6.37), Host:Debian6.04 (kernel 2.6.32.8)

Evaluation Questions

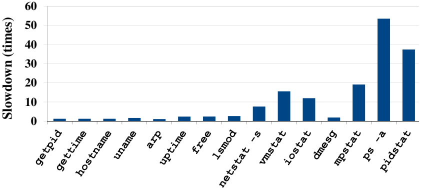
1. How fast our system really is?
2. HYBRID-BRIDGE vs. KVM
3. HYBRID-BRIDGE vs. VMST
4. HYBRID-BRIDGE vs. VIRTUOSO
5. How often does the execution trap to SLOW-BRIDGE

FAST-BRIDGE Slowdown Compared to KVM

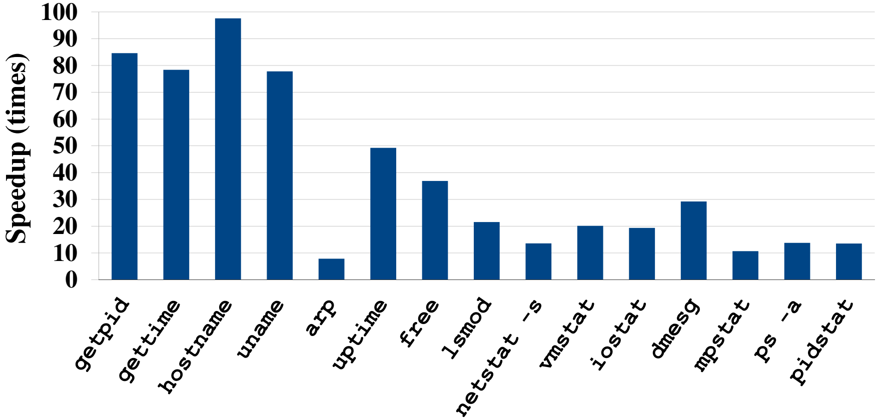


Fast-Bridge Slowdown Compared to KVM

App. Name	Hybrid-Bridge *#VMExit*	Slowdown Fast-Bridge vs. KVM
getpid	2	1.25X
gettime	4	1.25X
hostname	10	1.25X
uname	10	1.66X
arp	1852	1.09X
uptime	1892	2.40X
free	3927	2.42X
lsmod	11875	2.66X
netstat	23165	7.64X
vmstat	86578	15.57X
iostat	97390	12.00X
dmesg	11663	1.90X
mpstat	124525	19.12X
ps	418124	53.44X
pidstat	490713	37.37X

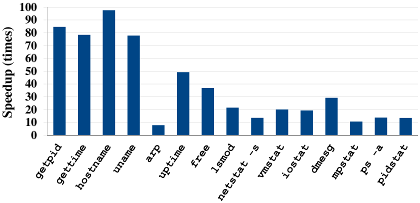


FAST-BRIDGE Speedup Compared to VMST



FAST-BRIDGE Speedup Compared to VMST

App. Name	HYBRID-BRIDGE *#VMExit*	Speedup FAST-BRIDGE vs. VMST
getpid	2	84.60X
gettime	4	78.40X
hostname	10	97.60X
uname	10	77.80X
arp	1852	7.86X
uptime	1892	49.25X
free	3927	36.88X
lsmod	11875	21.54X
netstat	23165	13.59X
vmstat	86578	20.13X
iostat	97390	19.35X
dmesg	11663	29.22X
mpstat	124525	10.68X
ps	418124	13.76X
pidstat	490713	13.53X

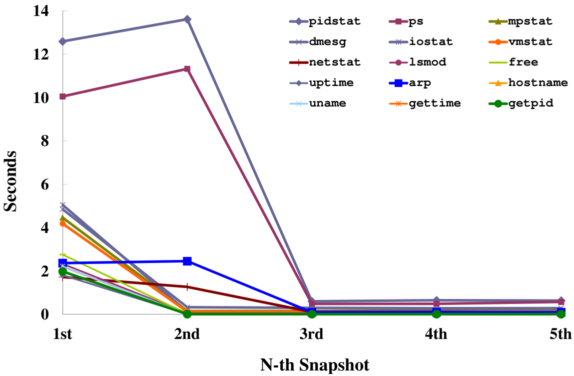


FAST-BRIDGE VS. VIRTUOSO

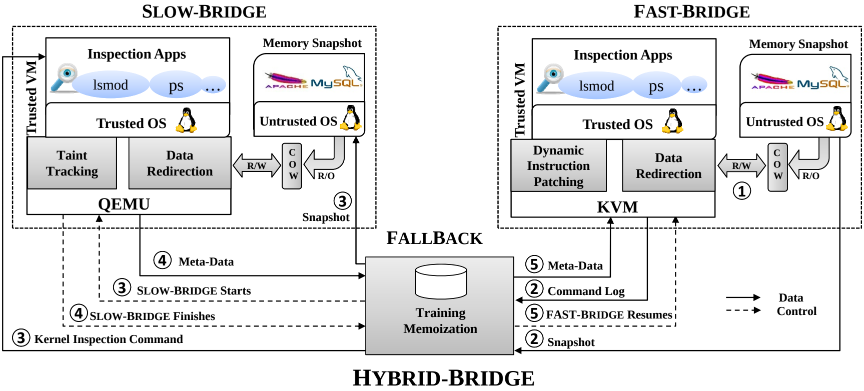
App. Name	Description	#X86 Inst. in VIRTUOSO	FAST-BRIDGE (sec.)	FAST-BRIDGE VS. VIRTUOSO
gettime	Tells current time of system	482	0.005	4.60X
getpid	Shows pid of current process	516	0.005	4.80X
tinyps	A compact version of PS	140843	0.064	23.45X
getprocname	Displays current Process Name	294797	0.132	20.57X

How often Hybrid-Bridge falls back to Slow-Bridge

App. Name	Hybrid-Bridge w/o any *MD* (sec.)	Hybrid-Bridge w/ Full *MD* (sec.) (i.e. Fast-Bridge)
getpid	1.976	0.005
gettime	1.985	0.005
hostname	2.199	0.005
uname	2.211	0.005
arp	2.360	0.094
uptime	1.810	0.012
free	2.755	0.017
lsmod	2.329	0.048
netstat	1.719	0.107
vmstat	4.186	0.109
iostat	5.047	0.120
dmesg	4.845	0.295
mpstat	4.460	0.153
ps	10.047	0.481
pidstat	12.585	0.598

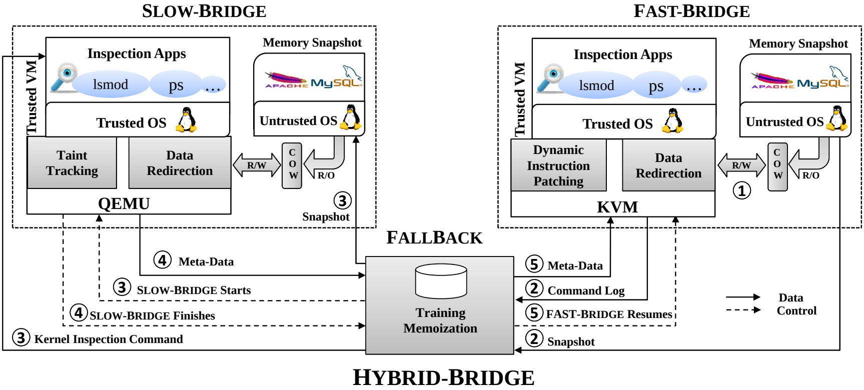


HYBRID-BRIDGE



- Combining the strength of both VIRTUOSO and VMST
- **Decoupling** the taint tracking component
- **Training memoization**

Thank you!



saberi.alireza@utdallas.edu
yangchun.fu@utdallas.edu
zhiqiang.lin@utdallas.edu