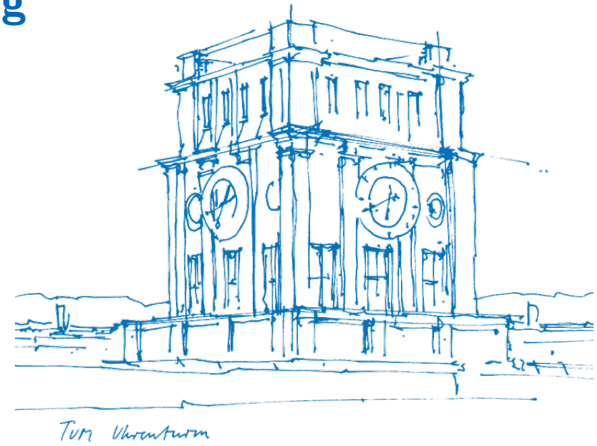
Chair of IT Security
School of Computation, Information and Technology
Technical University of Munich

HyperMirage: Direct State Manipulation in Hybrid Virtual CPU Fuzzing

Manuel Andreas,
Fabian Specht, Marius Momeu

Chair of IT Security
School of Computation, Information and Technology
Technical University of Munich

February 25th, 2026



Direct State Manipulation in Hybrid Virtual CPU Fuzzing

Motivation

Motivation

- Hypervisors form the cornerstone of security in cloud computing

Direct State Manipulation in Hybrid Virtual CPU Fuzzing

Motivation

- Hypervisors form the cornerstone of security in cloud computing
- Two major jobs:
  - Device virtualization (e.g., QEMU)
  - CPU virtualization (e.g., KVM)

Direct State Manipulation in Hybrid Virtual CPU Fuzzing

Motivation

- Hypervisors form the cornerstone of security in cloud computing
- Two major jobs:
  - Device virtualization (e.g., QEMU)
  - CPU virtualization (e.g., KVM)
- Lots of existing research exists on fuzzing/testing hypervisors, yet:

Direct State Manipulation in Hybrid Virtual CPU Fuzzing

Motivation

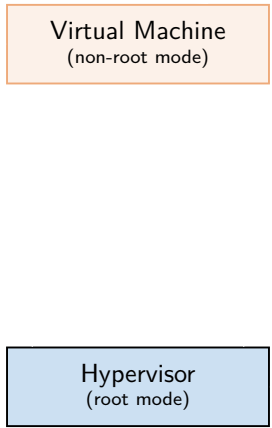
- Hypervisors form the cornerstone of security in cloud computing
- Two major jobs:
  - Device virtualization (e.g., QEMU)
  - CPU virtualization (e.g., KVM)
- Lots of existing research exists on fuzzing/testing hypervisors, yet:
  - Primarily focused on device virtualization (i.e., user space)

Direct State Manipulation in Hybrid Virtual CPU Fuzzing

Motivation

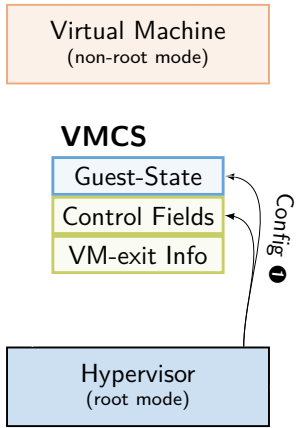
- Hypervisors form the cornerstone of security in cloud computing
- Two major jobs:
  - Device virtualization (e.g., QEMU)
  - CPU virtualization (e.g., KVM)
- Lots of existing research exists on fuzzing/testing hypervisors, yet:
  - Primarily focused on device virtualization (i.e., user space)
  - CPU virtualization received much less scrutiny (i.e., kernel space)

Intel VMX Rundown



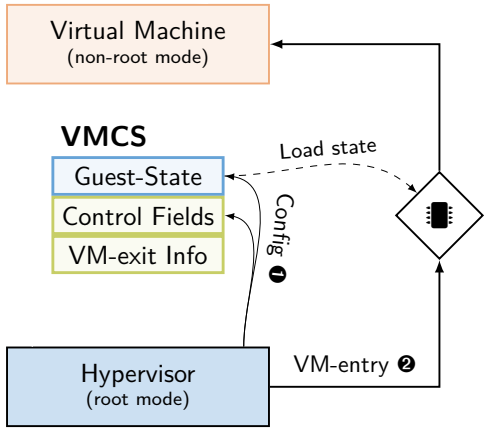
Direct State Manipulation in Hybrid Virtual CPU Fuzzing

Intel VMX Rundown



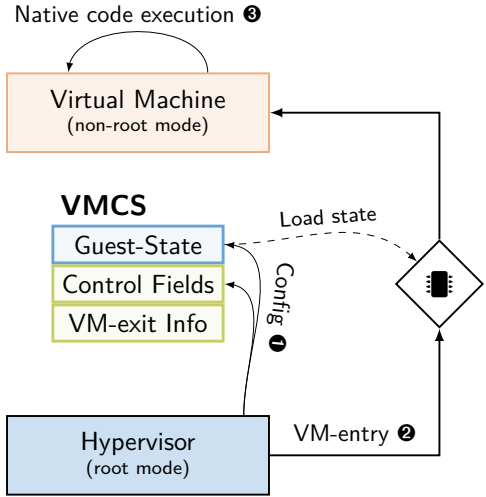
Direct State Manipulation in Hybrid Virtual CPU Fuzzing

Intel VMX Rundown



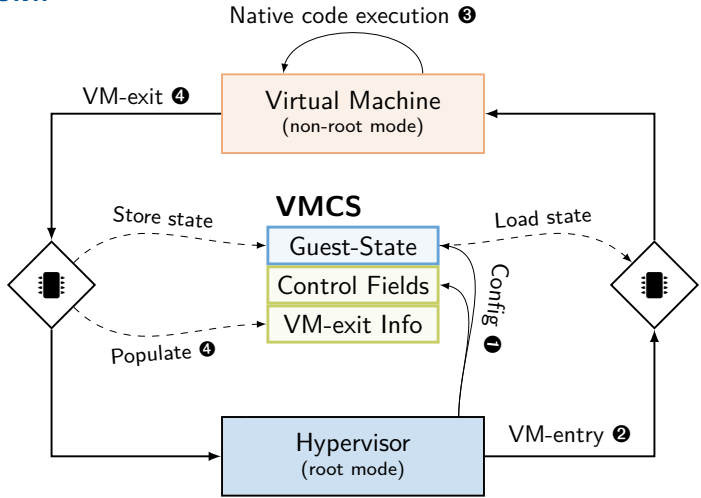
Direct State Manipulation in Hybrid Virtual CPU Fuzzing

Intel VMX Rundown



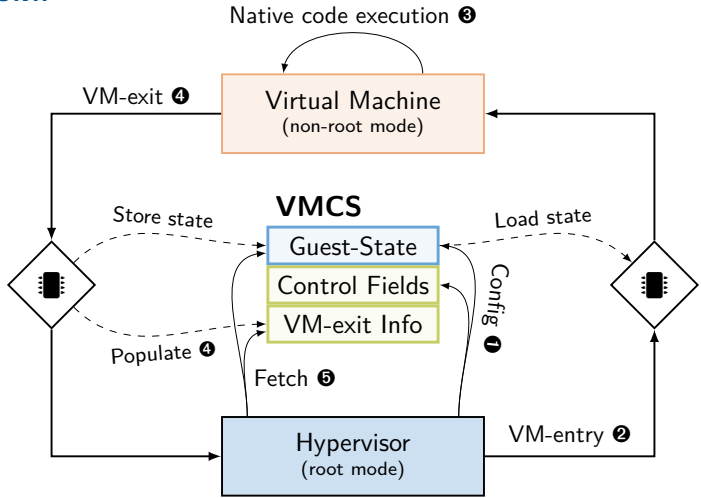
Direct State Manipulation in Hybrid Virtual CPU Fuzzing

Intel VMX Rundown



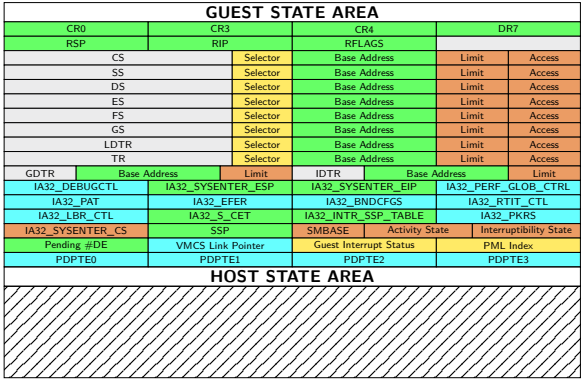
Direct State Manipulation in Hybrid Virtual CPU Fuzzing

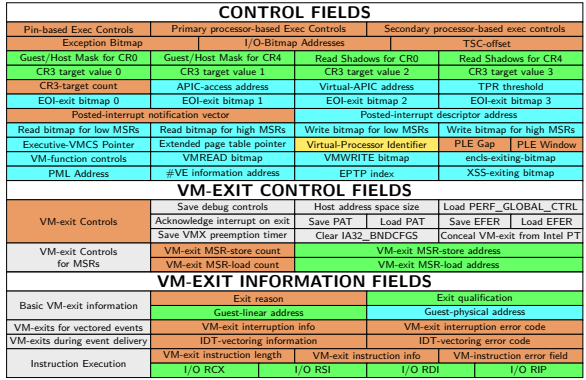
Intel VMX Rundown



Direct State Manipulation in Hybrid Virtual CPU Fuzzing

VMCS Fields





Natural-width field
16-bit field
32-bit field
64-bit field

VMCS — VM-Exit Information Fields

VM-EXIT INFORMATION FIELDS		
Basic VM-exit information	Exit reason	Exit qualification
	Guest-linear address	Guest-physical address
VM-exits for vectored events	VM-exit interruption info	VM-exit interruption error code
VM-exits during event delivery	IDT-vectoring information	IDT-vectoring error code

Natural-width field
16-bit field
32-bit field
64-bit field

Direct State Manipulation in Hybrid Virtual CPU Fuzzing

VM-exit Reasons

Exit Reason	Description
0	Exception or non-maskable interrupt (NMI)
1	External interrupt
2	Triple fault
3	INIT signal
4	Startup-IPI (SIPI)
5	I/O system-management interrupt (SMI)
6	Other SMI
7	Interrupt window
8	NMI window
9	Task switch
10	CPUID instruction
11	GETSEC instruction
12	HLT instruction
14	INVLPG instruction
15	RDPMC instruction
16	RDTSC instruction
17	RSM instruction in SMM
18 .. 27	VMX instruction
28	Control-register access
29	MOV DR instruction
30	I/O instruction
31	RDMSR instruction
32	WRMSR instruction

Exit Reason	Description
33	Entry failure due to invalid guest state
34	Entry failure due to MSR loading
36	MWAIT instruction
37	Monitor trap flag
39	MONITOR instruction
40	PAUSE instruction
41	Entry failure due to machine-check event
42	TPR below threshold
44	APIC access
45	Virtualized EOI
46	Access to GDTR or IDTR
47	Access to LDTR or TR
48	EPT violation
49	EPT misconfiguration
50	INVEPT instruction
51	RDTSCP instruction
52	VMX-preemption timer expired
53	INVVPID instruction
54	WBINVD or WBNOINVD instruction
55	XSETBV instruction
56	APIC write
57	RDRAND instruction
...	...

Direct State Manipulation in Hybrid Virtual CPU Fuzzing

VM-exit Handling

```
void vmx_vmexit_handler(void) {
    uint exit_reason = vmread(EXIT_REASON);
    switch (exit_reason) {
        case EXIT_HYPERCALL: {
            /* ... */
        }
        case EXIT_APIC_ACCESS: {
            /* ... */
        }
        case EXIT_MSR_READ: {
            /* ... */
        }
        /* ... */
    }
```

Direct State Manipulation in Hybrid Virtual CPU Fuzzing

VM-exit Handling

```
void vmx_vmexit_handler(void) {
    uint exit_reason = vmread(EXIT_REASON);
    switch (exit_reason) {
        case EXIT_HYPERCALL: {
            /* ... */
        }
        case EXIT_APIC_ACCESS: {
            /* ... */
        }
        case EXIT_MSR_READ: {
            /* ... */
        }
        /* ... */
    }
```

Direct State Manipulation in Hybrid Virtual CPU Fuzzing

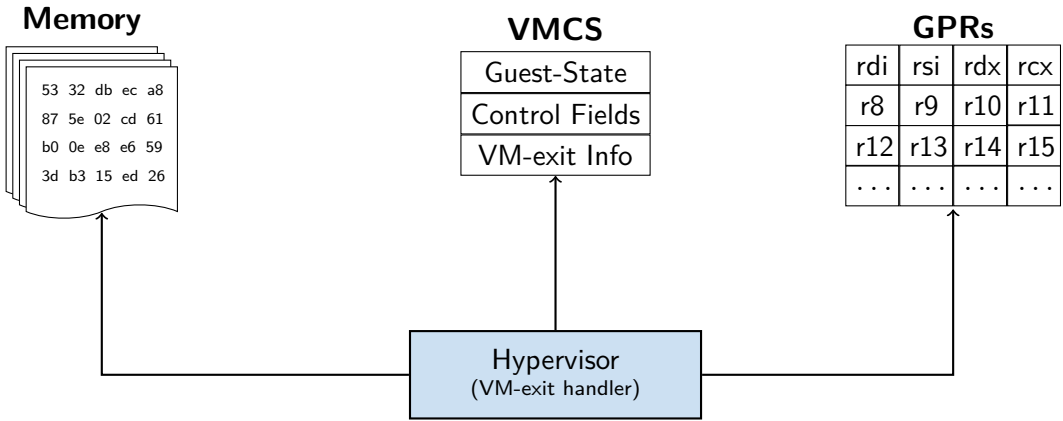
State-of-the-Art

Exit Reason	Description
0	Exception or non-maskable interrupt (NMI)
1	External interrupt
2	Triple fault
3	INIT signal
4	Startup-IPI (SIPI)
5	I/O system-management interrupt (SMI)
6	Other SMI
7	Interrupt window
8	NMI window
9	Task switch
10	CPUID instruction
11	GETSEC instruction
12	HLT instruction
14	INVLPG instruction
15	RDPMC instruction
16	RDTSC instruction
17	RSM instruction in SMM
18 .. 27	VMX instruction
28	Control-register access
29	MOV DR instruction
30	I/O instruction
31	RDMSR instruction
32	WRMSR instruction

Exit Reason	Description
33	Entry failure due to invalid guest state
34	Entry failure due to MSR loading
36	MWAIT instruction
37	Monitor trap flag
39	MONITOR instruction
40	PAUSE instruction
41	Entry failure due to machine-check event
42	TPR below threshold
44	APIC access
45	Virtualized EOI
46	Access to GDTR or IDTR
47	Access to LDTR or TR
48	EPT violation
49	EPT misconfiguration
50	INVEPT instruction
51	RDTSCP instruction
52	VMX-preemption timer expired
53	INVVPID instruction
54	WBINVD or WBNOINVD instruction
55	XSETBV instruction
56	APIC write
57	RDRAND instruction
...	...

Direct State Manipulation in Hybrid Virtual CPU Fuzzing

Relevant VM State



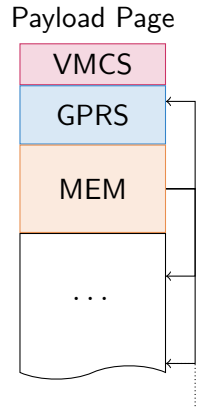
Overview

- Directly set all VM state the hypervisor sees during VM-exit handling
- Include all guest controlled parts of the VMCS, and:
  - Be able to fuzz all VM-exit reasons!
- Require no manual crafting of fuzzing seeds

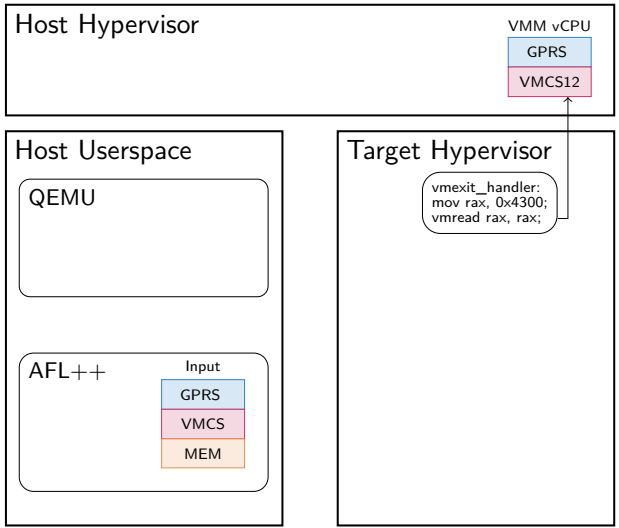
Input Modeling

Input split into 3 regions:

- VMCS:
  - Architectural State (CR0, CR3, EFER, LSTAR, ...)
  - Segments (CS, ES, DS, SS, ...)
  - VM-exit information:
    - Exit reason
    - Exit qualification
    - ...
- GPRs:
  - RDI, RSI, RDX, RCX, RBP, ...
- Memory:
  - 512 Bytes
  - Expanded to fill the entire page where the fuzzing input lies



Make the Hypervisor see a Mirage:

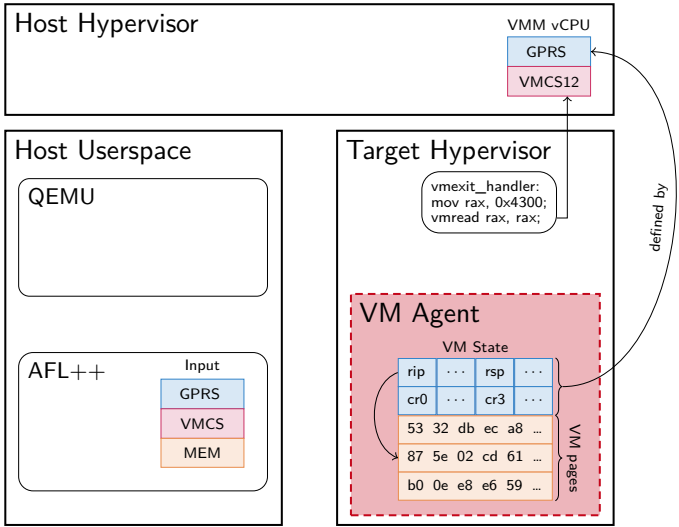


Direct State Manipulation in Hybrid Virtual CPU Fuzzing

Rundown

Make the Hypervisor see a Mirage:

- Spawn a dummy "VM Agent"

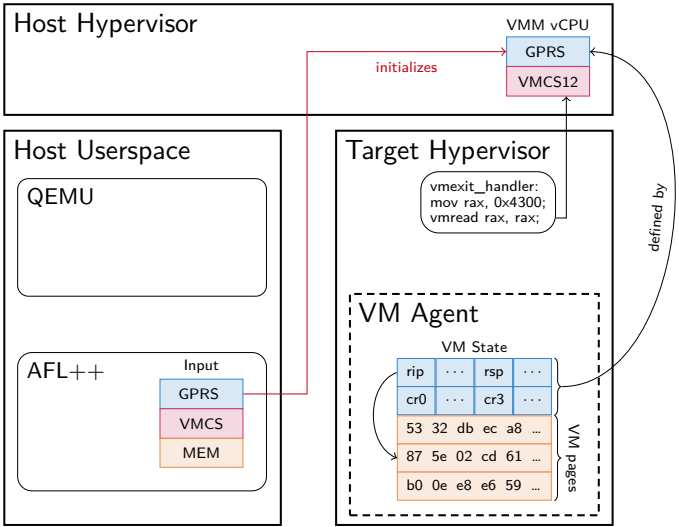


Direct State Manipulation in Hybrid Virtual CPU Fuzzing

Rundown

Make the Hypervisor see a Mirage:

- Spawn a dummy "VM Agent"
- Manipulate the VMCS according to fuzzing input

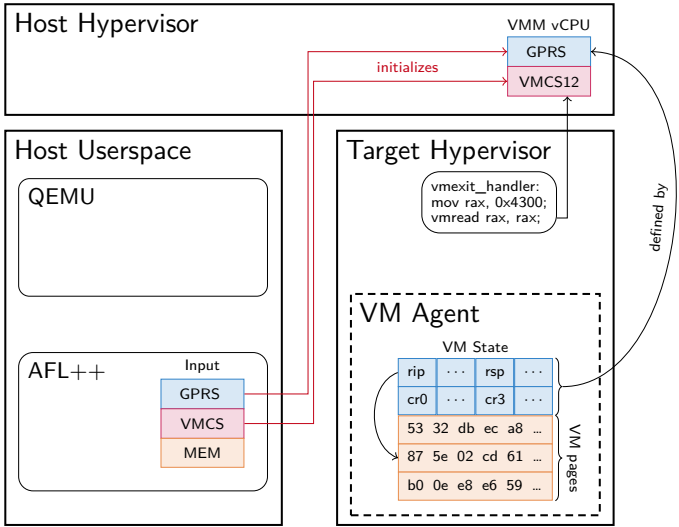


Direct State Manipulation in Hybrid Virtual CPU Fuzzing

Rundown

Make the Hypervisor see a Mirage:

- Spawn a dummy "VM Agent"
- Manipulate the VMCS according to fuzzing input
- Overwrite VM GPRs

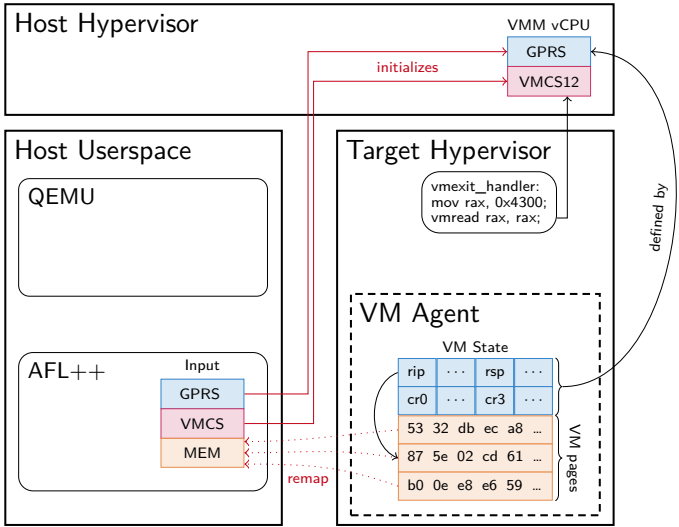


Direct State Manipulation in Hybrid Virtual CPU Fuzzing

Rundown

Make the Hypervisor see a Mirage:

- Spawn a dummy "VM Agent"
- Manipulate the VMCS according to fuzzing input
- Overwrite VM GPRs
- Remap all memory pages to the fuzzing input

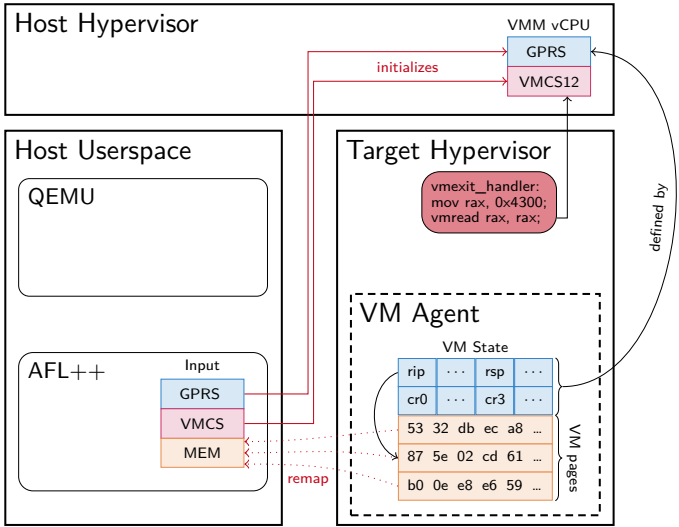


Direct State Manipulation in Hybrid Virtual CPU Fuzzing

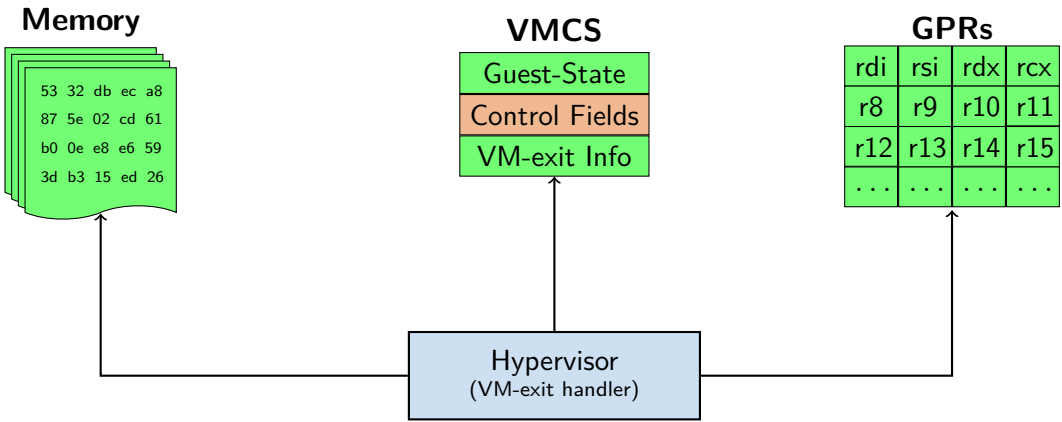
Rundown

Make the Hypervisor see a Mirage:

- Spawn a dummy "VM Agent"
- Manipulate the VMCS according to fuzzing input
- Overwrite VM GPRs
- Remap all memory pages to the fuzzing input
- Inject VM-exit

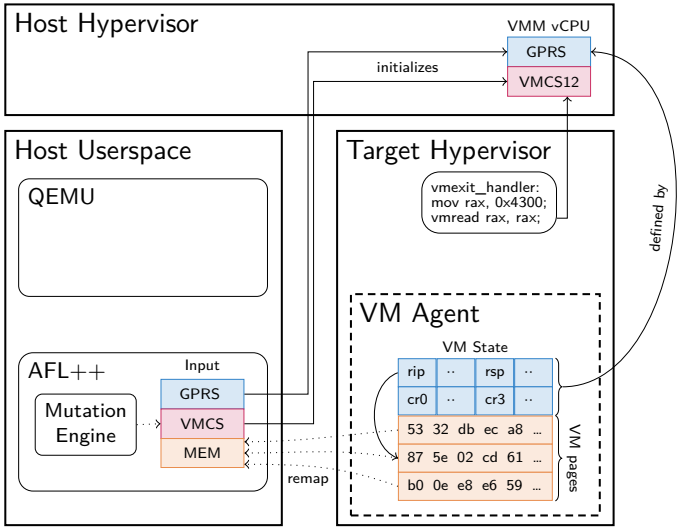


Direct State Manipulation in Hybrid Virtual CPU Fuzzing



Direct State Manipulation in Hybrid Virtual CPU Fuzzing

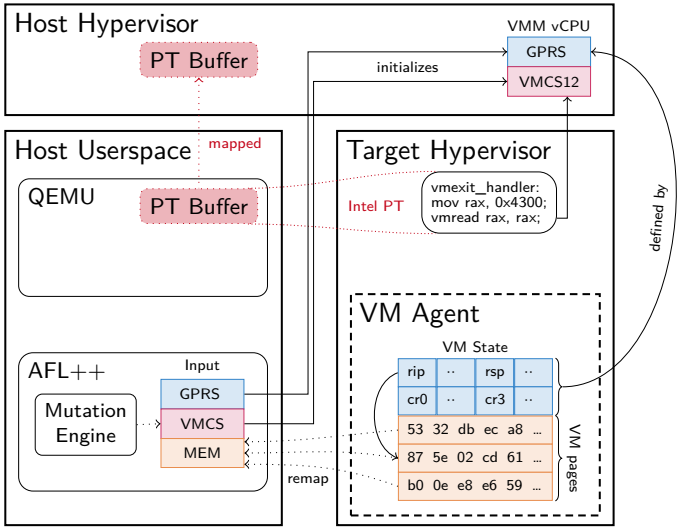
Overview



Direct State Manipulation in Hybrid Virtual CPU Fuzzing

Overview

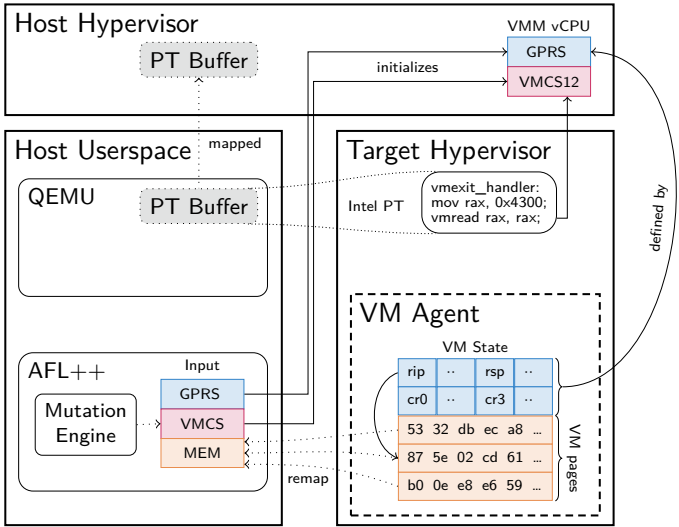
- Coverage feedback via Intel PT



Direct State Manipulation in Hybrid Virtual CPU Fuzzing

Overview

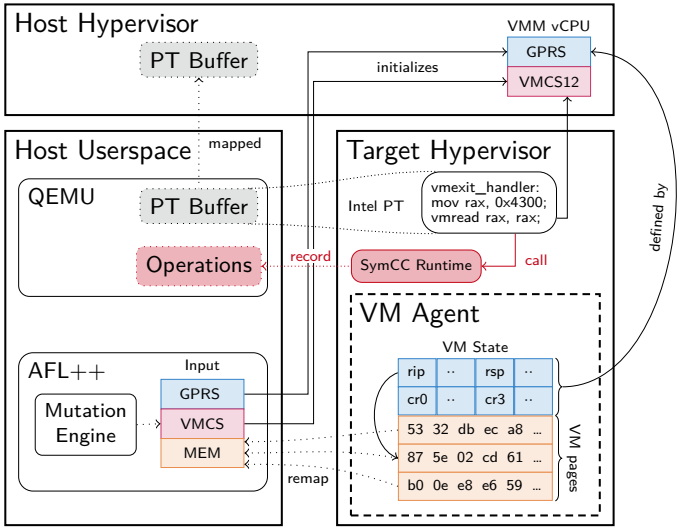
- Coverage feedback via Intel PT
- Symbolic execution based on SymCC [2]
  - Compiler-based instrumentation



Direct State Manipulation in Hybrid Virtual CPU Fuzzing

Overview

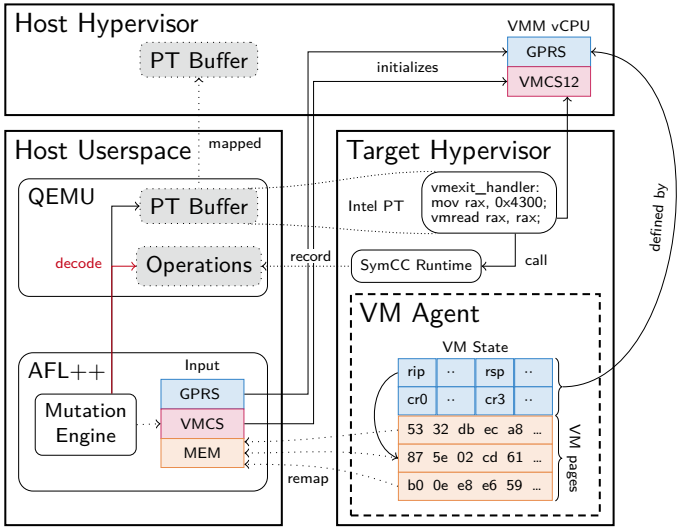
- Coverage feedback via Intel PT
- Symbolic execution based on SymCC [2]
  - Compiler-based instrumentation
- We extend SymCC to support bare-metal environments
  - Novel record-and-replay runtime



Direct State Manipulation in Hybrid Virtual CPU Fuzzing

Overview

- Coverage feedback via Intel PT
- Symbolic execution based on SymCC [2]
  - Compiler-based instrumentation
- We extend SymCC to support bare-metal environments
  - Novel record-and-replay runtime
  - Symbolic computations shifted outside of target



Direct State Manipulation in Hybrid Virtual CPU Fuzzing

Record-and-replay

- High throughput due to native execution
- Enables application of SymCC to bare-metal targets
- Systematically explore VM-exit handling code

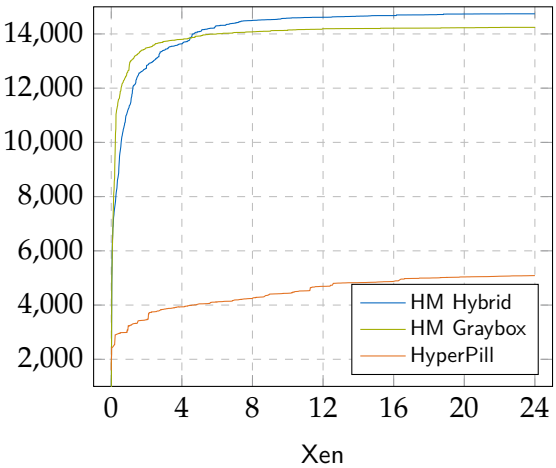
```
SymID _sym_build_add(SymID a, SymID b) {
  if (unlikely(!trace_enabled))
    return;

  submit_operation(OP_ADD);
  submit_value(a);
  submit_value(b);
  /* Represents symbolic return value */
  submit_value(placeholder_id++);
}
```

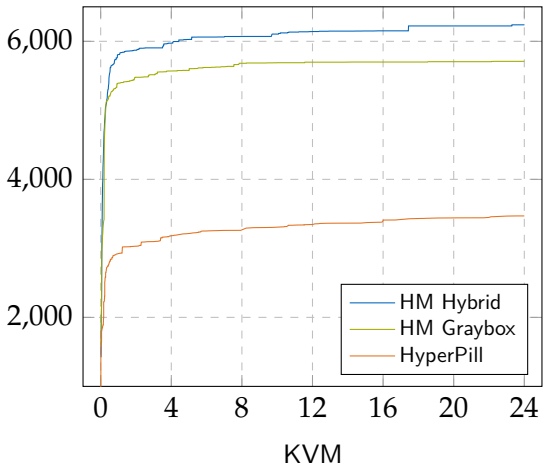
Evaluation
Setup

- Prototype of HyperMirage for Intel VMX platform
- Targeting:
  - Xen 
  - KVM 
- Comparison to HyperPill [1]

Evaluation

RQ1: HyperMirage vs. State-of-the-Art — Edge Coverage (24h)





Evaluation

RQ1: HyperMirage vs. State-of-the-Art — Performance (24h)

	HyperMirage		HyperPill
	Hybrid	**Graybox**	
Xen	3357	3720	23
KVM	2023	2277	93

Average executions per second by HyperMirage and HyperPill.

Evaluation

RQ2: Per VM-exit Coverage (24h)

	EPT VIOLATION	VMCALL	MSR WRITE	MSR READ	ACCESS LDTR OR TR	IO INSTRUCTION	VMLAUNCH	APIC ACCESS	RDTSCP	CPUID	VMXOFF	TASK SWITCH	CR ACCESS	INVPCID	GETSEC	MCE DURING VMENTER	TRIPLE FAULT	VMXON	PAUSE INSTRUCTION	HLT	XSETBV	VMPTRLD	VMREAD	EPT MISCONFIG	EXTERNAL INTERRUPT	INVEPT	INVLPG	VMWRITE	VMCLEAR	INVVPID	VMRESUME	VMPTRST	DR ACCESS	XRSTORS	NOTIFY	INVD	MSR LOADING	PENDING VIRT INTR	RDTSC	VMX TIMER EXPIRED	VMFUNC	INIT	...
Xen	8086	2971	458	323	235	231	178	136	124	119	112	111	78	●	❍	72	43	42	40	26	23	21	20	20	14	13	12	12	11	8	7	5	5	❍	5	4	4	❍	3	●	❍	❍	...
KVM	2365	222	1531	483	0	62	1	67	❍	64	17	173	182	117	●	❍	❍	6	2	60	17	21	13	44	❍	47	7	12	24	39	12	22	30	❍	12	❍	●	13	❍	4	0	●	...

● VM-exit not implemented by the HV.

❍ VM-exit handler that does not exhibit coverage.

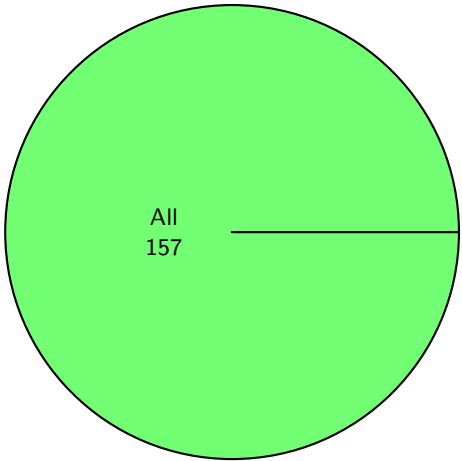
Evaluation
RQ3: Bug Discovery

CVE	Commit	Hypervisor	Description	VM-exit	Component	Signal
CVE-2023-46842	1166467e	Xen	HVM hypercalls may trigger Xen bug check.	VMCALL	Hypercall	Panic
N/A	d980886f	Xen	Out-of-bounds shift in memory exchange hypercall.	VMCALL	Hypercall	UBSan
CVE-2024-45818	c41c3d8c	Xen	Deadlock in HVM standard VGA handling.	APIC Emul.	MMIO	Panic
N/A	672894a1	Xen	MMIO cache emulation failure.	APIC Emul.	MMIO	Assert
N/A	N/A	Xen	MMIO cache emulation failure[r].	APIC Emul.	MMIO	Assert
N/A	59e6ad65	Xen	Missing cleanup on HVM memory mappings.	EPT Violation	Instruction Emul.	Assert
N/A	a677964c	Xen	Incorrect offset in instruction emulation.	EPT Violation	Instruction Emul.	Assert
N/A	73570ceb	Xen	Incorrect IP rollback in instruction emulation.	EPT Violation	Instruction Emul.	Assert
N/A	a150ecce	Xen	Out-of-bounds shift in FPU emulation.	EPT Violation	Instruction Emul.	UBSan
CVE-2025-38351	fa787ac0	KVM	INVVPID failure during PV TLB flush.	VMCALL	Hypercall	Panic
CVE-2025-38469	5a53249d	KVM	Memory leak in schedop poll hypercall.	VMCALL	Hypercall	Kmemleak

[r] Rediscovered after a fixing commit was released.

Evaluation
RQ4: False Positive Analysis

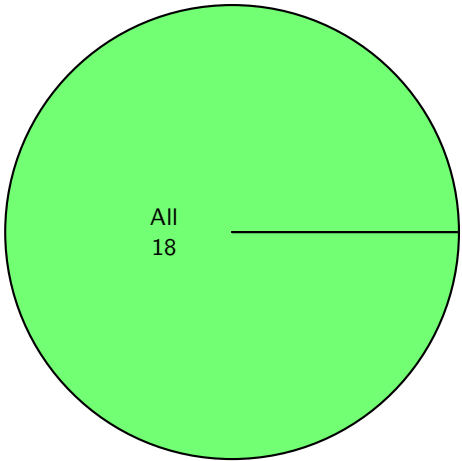
- 24-hour fuzzing run targeting Xen
- 157 unique discovered crashes



Evaluation

RQ4: False Positive Analysis

- 24-hour fuzzing run targeting Xen
- 157 unique discovered crashes
- After crash reason de-duplication: 18 unique crashes

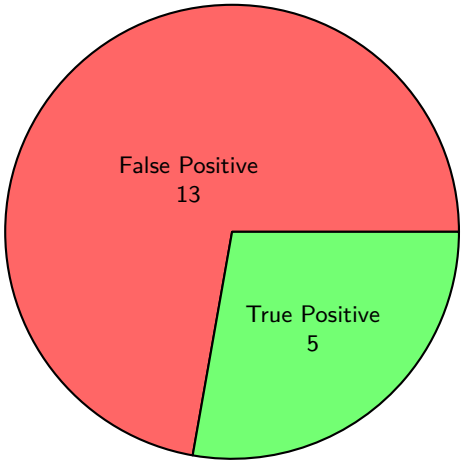


De-duplicated Crashes

Evaluation

RQ4: False Positive Analysis

- 24-hour fuzzing run targeting Xen
- 157 unique discovered crashes
- After crash reason de-duplication: 18 unique crashes
- Manual analysis yields:
  - 5 true positives
  - 13 false positives (i.e., architecturally invalid)

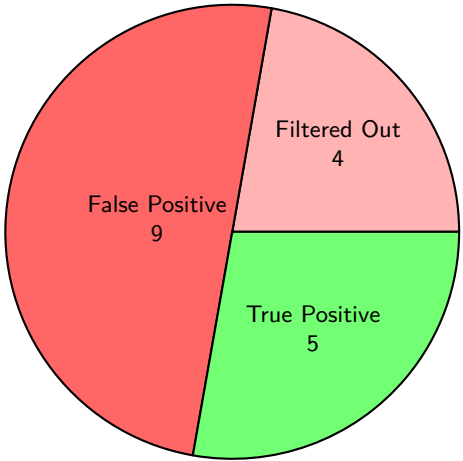


De-duplicated Crashes

Evaluation

RQ4: False Positive Analysis

- 24-hour fuzzing run targeting Xen
- 157 unique discovered crashes
- After crash reason de-duplication: 18 unique crashes
- Manual analysis yields:
  - 5 true positives
  - 13 false positives (i.e., architecturally invalid)

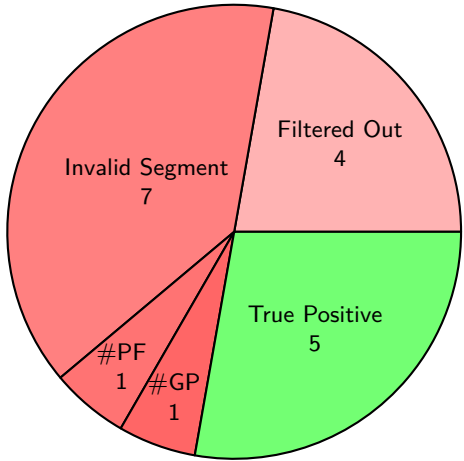


De-duplicated Crashes

Evaluation

RQ4: False Positive Analysis

- 24-hour fuzzing run targeting Xen
- 157 unique discovered crashes
- After crash reason de-duplication: 18 unique crashes
- Manual analysis yields:
  - 5 true positives
  - 13 false positives (i.e., architecturally invalid)



De-duplicated Crashes

Conclusion

- HyperMirage scrutinizes the majority of the virtual CPU attack surface
- High throughput due to native execution
- Record-and-replay runtime enables symbolic execution on bare-metal targets
- 11 novel bugs in battle-tested Hypervisors, including 4 CVEs
- Produces false positive crashes
  - Can be triaged in reasonable time
- Open-sourced prototype: `https://github.com/tum-itsec/hypermirage`

References

[1] A. Bulekov, Q. Liu, M. Egele, and M. Payer. "HYPERPILL: Fuzzing for Hypervisor-bugs by Leveraging the Hardware Virtualization Interface". In: *33rd USENIX Security Symposium (USENIX Security 24)*. 2024.

[2] S. Poeplau and A. Francillon. "Symbolic execution with SymCC: Don't interpret, compile!" In: *29th USENIX Security Symposium (USENIX Security 20)*. 2020.

Questions?

Read the paper!