Intrusion Injection for Virtualized Systems Concepts and Approach

Charles Gonçalves, Nuno Antunes, Marco Vieira

University of Coimbra

charles@dei.uc.pt

1290 FACULDADE DE CIÊNCIAS E TECNOLOGIA UNIVERSIDADE D COIMBRA

fct Fundação para a Ciência e a Tecnologia

REPÚBLICA PORTUGUESA CIÊNCIA, TECNOLOGIA E ENSINO SUPERIOR

Cofinanciado por: CENTRO 2020 PORTUGAL 2020 UNIÃO EUROPEIA Fundo Social Europeu

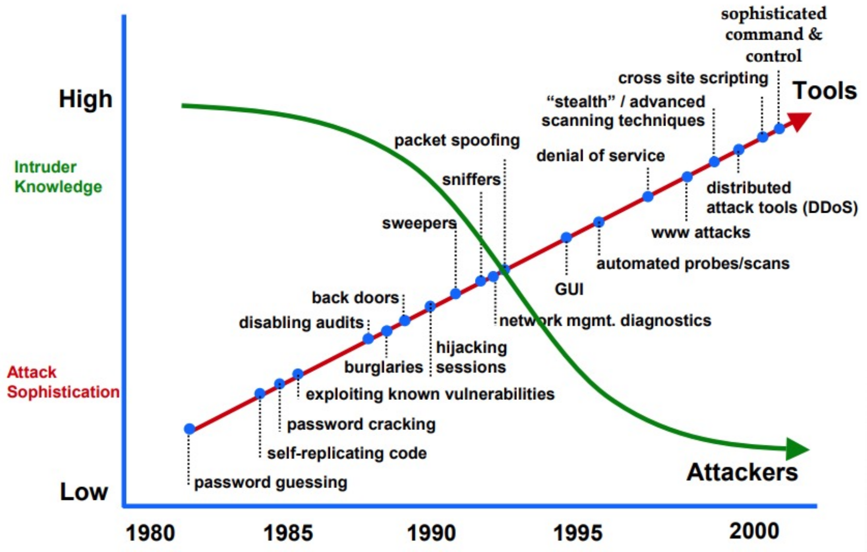
CEFET-MG

Motivation

- Cloud Computing: *main motivation*
  - *"The use of cloud computing has been growing in the last few years…"*
- Virtualization
  - Keystone technology that enables cloud
- Has been adopted across various industries to enhance efficiency, flexibility, and cost-effectiveness, including **safety-critical** areas
  - Aviation, automotive, healthcare, nuclear power plants…

Contextualization

- Hypervisors:
  - Manage the physical resources
  - Complex and undoubtedly have bugs
- *Limited approaches to assessing its security*
- Security mechanism can't avoid exploitation
- *The current dilemma is "* Not *how,* but *when"*



Source: Lipson, F. H, Tracking and Tracing Cyber-Attacks: Technical Challenges and Global Policy Issues

Evaluating Security

- Lack of consolidated and practical solutions
- Rely on vulnerabilities to this goal is unreasonable:
  - Impossible to predict (vulnerability/attack)
  - Attack a real vulnerability can be a challenge
  - Many vulnerabilities were fixed prior to the release
  - Vulnerabilities should be fixed
- Create a representative attack corpus seems impossible

Problem

Understand how **hypervisors** deal with **successful intrusions**, even if we don’t know :

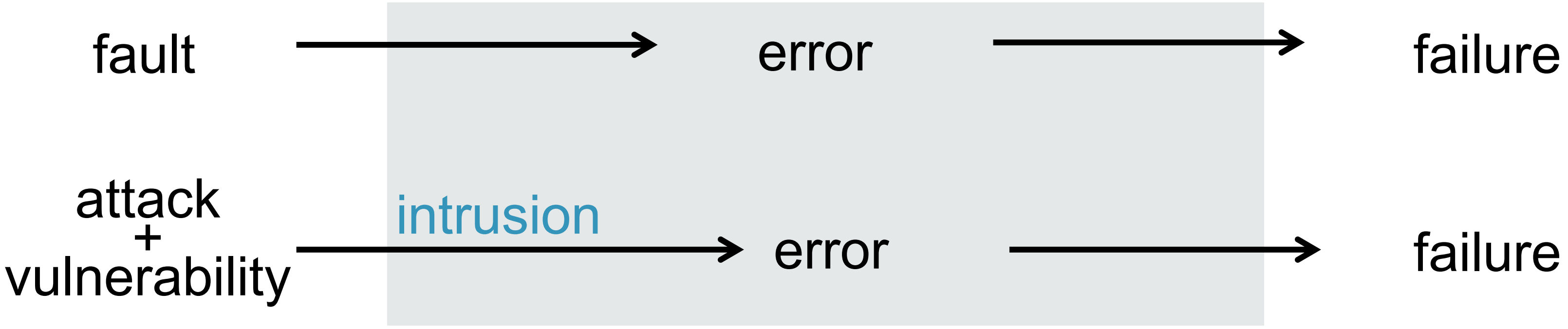
- What vulnerabilities are in the system
- Which attacks may occur

Agenda

- Concepts
- Intrusion Injection
- Feasibility test
- Conclusions and Limitations

Concepts

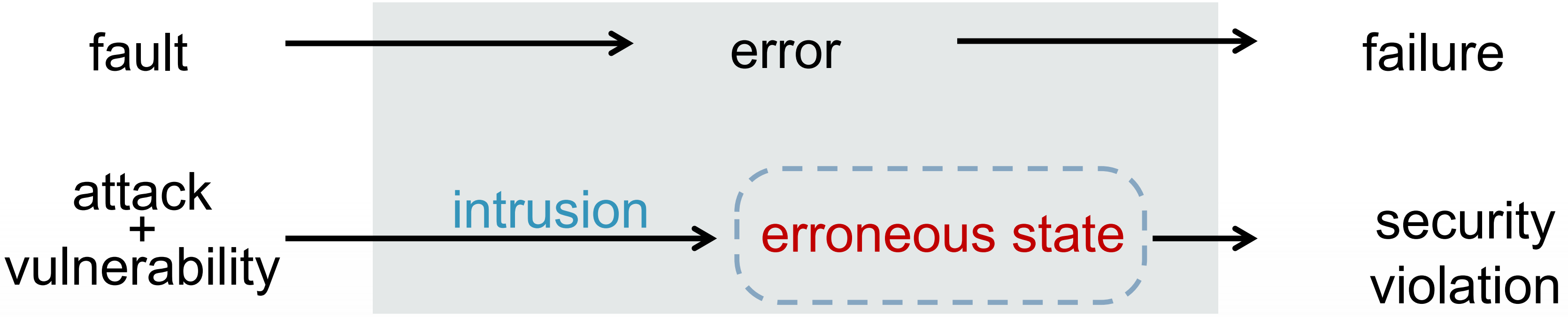
AVI model



Intrusion - a malicious operational fault resulting from a successful attack on a vulnerability

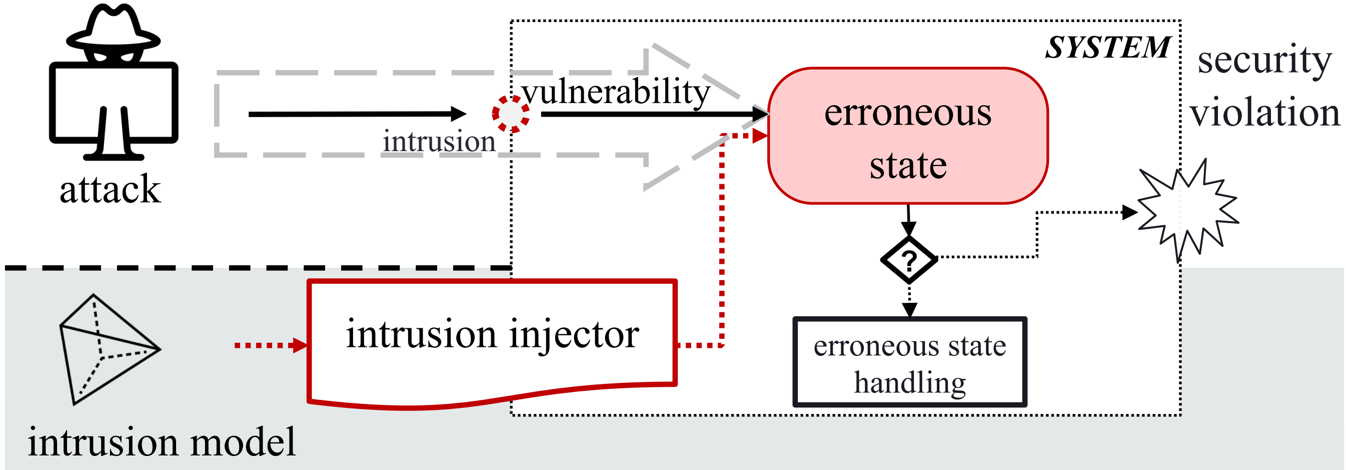
Verissimo, Neves, and Correia, “Intrusion-Tolerant Architectures: Concepts and Design,” in *Architecting Dependable Systems*, Springer, Berlin, Heidelberg, 2003, pp. 3–36.

Extending the AVI model



erroneous states are used instead of errors to emphasize that these are intrusion induced errors

Intrusion Injection

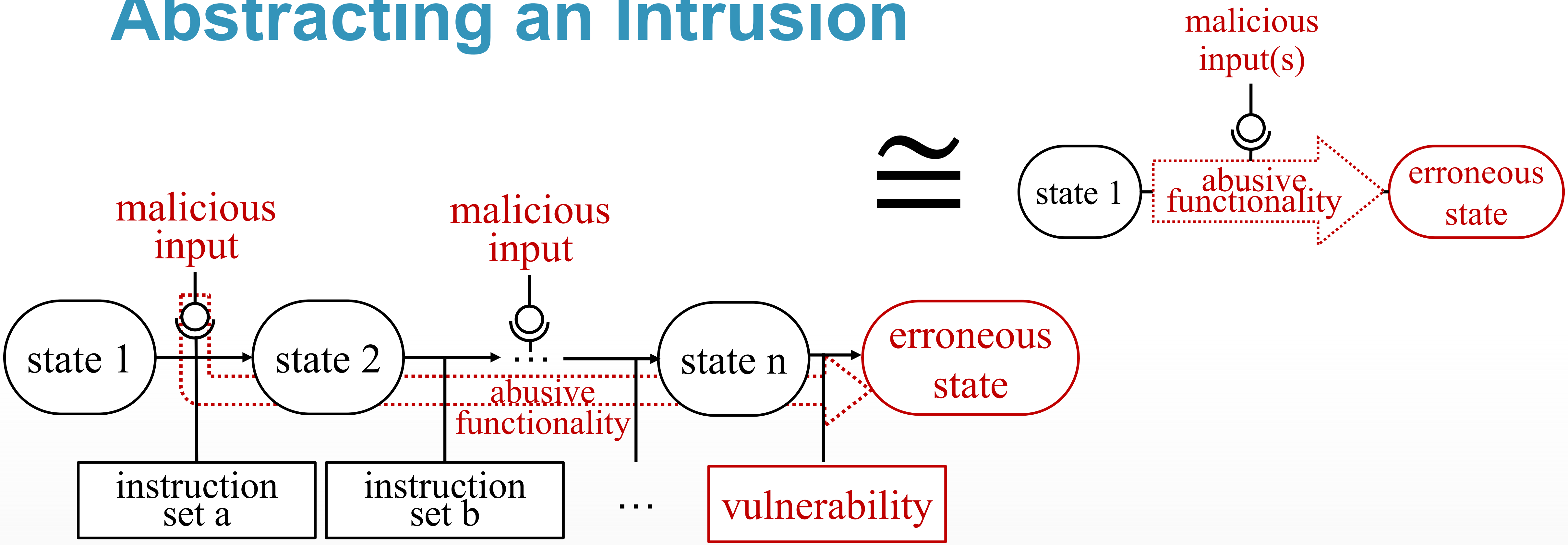


In the same way **errors** can be injected to emulate the effects of potential faults, they can be injected to mimic the effect of attacks on potential vulnerabilities

Intrusion Model (IM)

- The definition of the main aspects of the injection
- Crucial to achieve representativeness
  - Security implications
- It must be closely related to attacks and should generalize the main concepts
  - Attack and erroneous states
- But how to model that !?
  - Sergey Bratus : "*What hacker research taugh me*"
    - One intended machine endless ***Weird Machines***

Abstracting an Intrusion



IM Definition

- The intrusion Model ***abstracts*** how an erroneous state is achieved when using an abusive functionality through a given interface
- XSA-387: bug in grant table
- XSA-393: bug decrese_reservation hypercall
  - Very different bug
  - Exact abusive functionality: allows a guest to keep access to a memory page after a release operation

Abusive functionality

- We assume the existence of a set of AF capable of generalizing the outcome of different classes of intrusions
- Initial investigation ~100 CVE shows evidence that this might be possible

TABLE I
EXAMPLE OF ABUSIVE FUNCTIONALITIES THAT CAN BE OBTAINED FROM ACTIVATING XEN VULNERABILITIES.

Memory Access – 35 CVEs	
Read Unauthorized Memory	11
Write Unauthorized Memory	09
Write Unauthorized Arbitrary Memory	02
R/W Unauthorized Memory	12
Fail a Memory Access	01
Memory Management – 40 CVEs	
Corrupt Virtual Memory Mapping	04
Corrupt a Page Reference	04
Decrease Page Mapping Availability	01
Guest-Writable Page Table Entry	07
Fail a memory mapping	02
Uncontrolled Memory Allocation	11
Keep Page Access	11
Exceptional Conditions – 11 CVEs	
Induce a Fatal Exception	06
Induce a Memory Exception	05
Non-Memory Related – 22 CVEs	
Induce a Hang State	20
Uncontrolled Arbitrary Interrupts Requests	02

The Reachability Problem

- A particular state can be reached through a specific sequence of events: i.e, *an erroneous state can be reached from an intrusion injection technique*
- Challenges:
  - ***Security-related EE vs accidental faults EE***
  - Reachable by Intrusion Injection but not by any exploitation
  - Technical feasibility of certain erroneous states
  - Unknown erroneous states

Feasibility Test

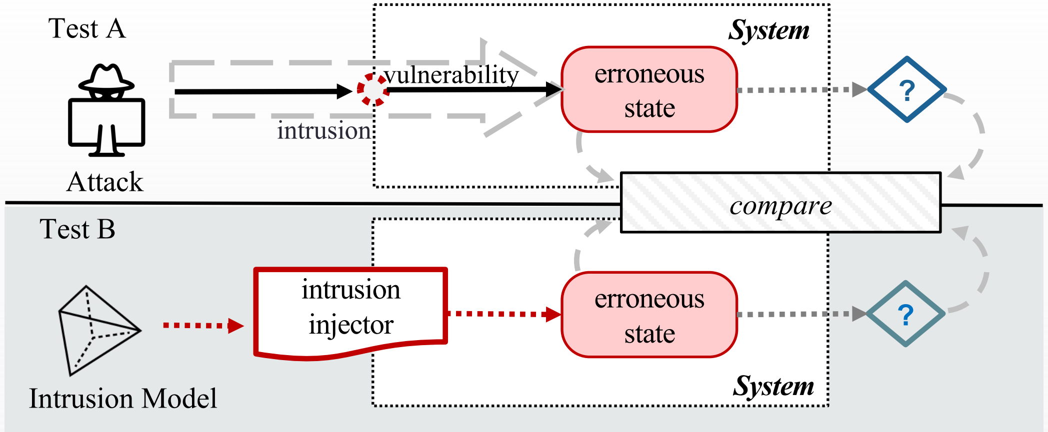
Memory Intrusion Injector Prototype

```
int do_arbitrary_access(
  unsigned long addr,   // Target Address
  void *buff,           // Buffer to read/write
  size_t n,             // Buffer size
  int action);          // Operation and Mode
```

- An Intrusion Injector that allows read/write in:
  - Hypervisor linear space
  - Physical Address
- Uses Xen directives to directly access the hardware
- Easily portable to different versions of Xen

Experimental Feasibility Test

- **RQ1**: Is it possible to inject erroneous states in a virtualized system in a way that **emulates the effects of** attacks exploiting real **vulnerabilities**?
- **RQ2**: Can intrusion injection induce the erroneous states (similar to those observed in real intrusions) in **nonvulnerable versions**?
- **RQ3**: Can intrusion injection **potentially support the assessment of security** attributes in virtualized systems?
- ***Reachability problem?***
  - i.e. realistic EE
  - Ground the experiments on exploits.



Case Studies

Use Case	Abusive Functionality
XSA-212-crash	Write Arbitrary Memory
XSA-212-priv	Write Arbitrary Memory
XSA-148-priv	Write Page Table Entries
XSA-182-test	Write Page Table Entries

```
-   ret = HYPERVISOR_memory_op(XENMEM_exchange, &args);
+   ret = HYPERVISOR_arbitrary_access(
+           args.out.extent_start + 8 * args.nr_exchanged,
+        &content,
+           sizeof(u64),
+           ARBITRARY_WRITE_LINEAR
+
+   );
```

XSA-212

- Vulnerability in `memory_exchange()` hypercall
- Enable the **arbitrary memory access** by encoding target address as input

XSA-212-crash (RQ1 and RQ2)

- Goal: Crash the host
- Overwrite the page fault handler in IDT and cause a double fault
- Erroneous State: corrupted page fault handler

```
(XEN) *** DOUBLE FAULT ***
(XEN) ----[ Xen-4.6.0  x86_64  debug=n  Tainted:    C ]----
(XEN) CPU:    23
(XEN) RIP:    e033:[<000000000042045d>] 000000000042045d
(XEN) RFLAGS: 0000000000010202   CONTEXT: hypervisor
...
(XEN) ***********************************
(XEN) Panic on CPU 23:
(XEN) DOUBLE FAULT -- system shutdown
(XEN) ***********************************
(XEN)
(XEN) Reboot in five seconds...
```

Use Case	4.6		4.8		4.13	
	PoC	Effector	PoC	Effector	PoC	Effector
Erroneous State	✓	✓	✗	✓	✗	✓
Security Violation	✓	✓	n/a	✓	n/a	✓

XSA-212-priv (RQ3)

- Goal: Privilege Escalation
- Create a writable page table for a hidden memory location and install an exploit
- Erroneous State: writable page table

```
[  116.268081] ### crafted PUD entry written
[  116.284080] going to link PMD into target PUD
[  116.292081] linked PMD into target PUD
```

```
root@guest03 ~/xsa212/privesc_poc:
$ ./attack 'echo "|$(id)|@$hostname)"' > /tmp/injector_log
press enter to contine

root@guest03 ~/xsa212/privesc_poc:
```

```
root@xen3 ~:
$ cat /tmp/injector_log
|uid=0(root) gid=0(root) groups=0(root)|@xen3
```

Use Case	4.6 PoC	4.6 Effector	4.8 PoC	4.8 Effector	4.13 PoC	4.13 Effector
Erroneous State	✓	✓	✗	✓	✗	✓
Security Violation	✓	✓	n/a	✓	n/a	⛉

Conclusions and Limitations

- It can **emulate the effects** of exploiting real vulnerabilities, even in **nonvulnerable** version and we have some **evidence that it can support the assessment of security**
- **Easier** than developing exploits/tools to attack the system
- Possibly study the impact of **unknown** vulnerabilities
- Portable test cases based on **architecture** of the target systems rather than on **implementation**

- IMs are not yet well defined and need further investigation
- The reachability problem is still an open issue

Questions ?

Charles Gonçalves, Nuno Antunes, Marco Vieira

University of Coimbra

charles@dei.uc.pt

Funding:

- *Project “Agenda Mobilizadora Sines Nexus*
- *FCT - Foundation for Science and Technology*
- *SPEC RG Security Benchmarking*
- *CEFET-MG*

Cofinanciado por: