HInjector: Injecting Hypercall Attacks for Evaluating VMI-based Intrusion Detection Systems

Aleksandar Milenkoski[1], Bryan D. Payne[2], Nuno Antunes[3], Marco Vieira[3], Samuel Kounev[1]
[1] Karlsruhe Institute of Technology, Karlsruhe, Germany
{milenkoski, kounev}@kit.edu
[2] Nebula Inc., CA, USA
bdpayne@acm.org
[3] University of Coimbra, Coimbra, Portugal
{nmsa, mvieira}@dei.uc.pt

Motivation, Scope, and Approach

› VMI (virtual machine introspection) is a mechanism for monitoring states of guest VMs (virtual machines) from a virtualization host

› VMI is used for attacker-transparent intrusion detection in virtualized environments

› Problem: Evaluation of the attack detection accuracy of VMI-based intrusion detection systems for detecting attacks targeting VMMs (virtual machine monitors)

› Attack vectors: device drivers, VM exit events, hypercalls

› Low number of publicly available attack scripts demonstrating hypercall attacks

› Solution: Automated artificial injection of malicious hypercalls with respect to representative attack models

› VMM of choice: Xen

Attack Models

› Attack models based on analyzing publicly available reports on vulnerabilities of Xen's hypercall handlers:

(i) invoking hypercalls from irregular call sites

(ii) invoking hypercalls with anomalous parameter values

› outside valid value domains

› specifically crafted for exploiting specific vulnerabilities (not necessarily outside valid value domains)

(iii) invoking a series of hypercalls in irregular order including repetitive execution of a single or multiple hypercalls

› Analyzed reports on vulnerabilities of Xen's hypercall handlers (selection): CVE-2008-3687, CVE-2012-3516, CVE-2012-5513, CVE-2012-6035, CVE-2013-1920

Configuration

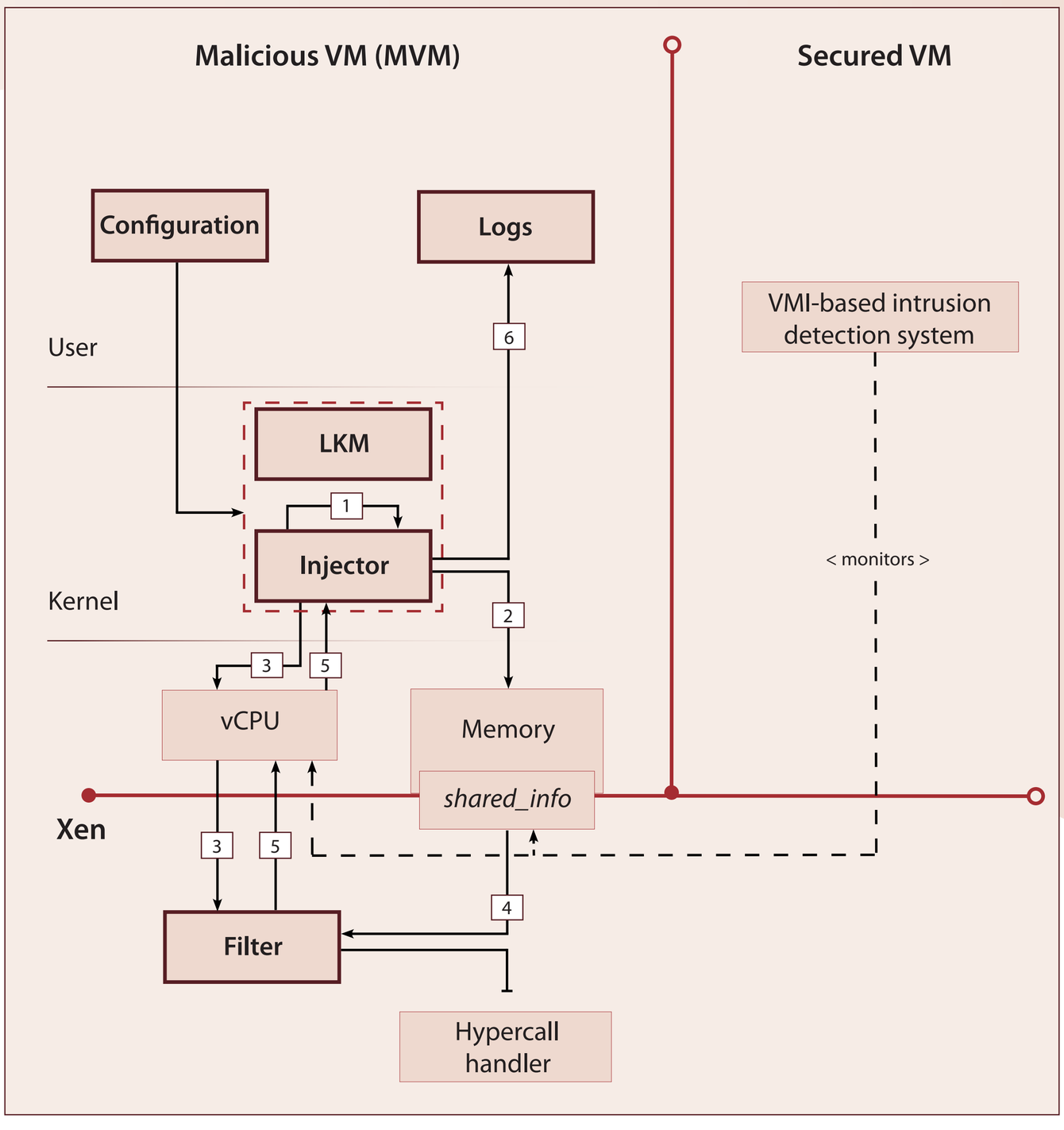
› User files containing configuration parameters:
› duration of an injection campaign
› valid parameter value domains
› specifically crafted parameter values
› valid order of hypercalls
› average injection rate and temporal distribution of injection actions

Injector

› Component deployed in the hypercall interface of the kernel of MVM
› Tasks:
› intercepts hypercalls invoked by the kernel
› modifies hypercall parameter values
› Used for injecting malicious hypercalls invoked from a regular call site

LKM

› A kernel module loaded in the kernel of MVM
› Tasks:
› invokes regular hypercalls
› invokes hypercalls with anomalous parameter values
› invokes series of hypercalls in irregular order
› Used for injecting hypercalls from an irregular call site



Logs

› User files containing records about injected hypercalls:
› hypercall identification numbers
› hypercall parameter values
› timestamps
› Used as "ground truth" information

Filter

› Component deployed in Xen's hypercall interrupt handler
› Tasks:
› identifies injected hypercalls
› blocks execution of injected hypercalls
› returns valid error codes

Injection of a Hypercall: An Example

› Injection of a hypercall with an anomalous parameter value by the Injector:

1) the Injector intercepts a hypercall invoked by the kernel of MVM and modifies the value of one of its parameters
2) the Injector stores the ID of the hypercall, the number of the parameter with anomalous value, and the parameter value in *shared_info*
3) the Injector passes the hypercall to the virtual CPU (vCPU) of MVM, which then passes execution control to Xen
4) the Filter reads the data stored in *shared_info*, identifies the injected hypercall, and blocks its execution
5) the Filter returns a valid error code
6) the Injector stores in the Logs the identification number and the parameter values of the injected hypercall, and a timestamp

Future Work

› Definition of representative characteristics of hypercall attacks:
› parameter values
› orders of hypercalls
› Provisioning of readily available configuration files for injecting representative hypercall attacks

› Challenge: Lack of publicly available technical information on vulnerabilities of Xen's hypercall handlers and hypercall attacks performed in practice