Cyber Experimentation of the Future (CEF)

Panel Discussion

Dr. Jinpeng Wei
Florida International University
Miami, FL, USA

The 31st Annual Computer Security Applications Conference (ACSAC), December 10th, 2015
Systems Software

- Applications
- Middleware
- Operating System
- Virtual Machine Manager (VMM)
My Research on Systems Software Security

- Runtime integrity of systems software
- Malware analysis, detection, and defense
- Secure software architecture
- Software vulnerability modeling, detection, risk-assessment, and prevention
- Security in cloud computing (e.g., MapReduce and Web Service Platforms)
- More ...
Hypothesis: a rootkit has to violate the integrity of the victim OS to some extent
- E.g., a rootkit tampers with the system call table to hide its files from user-space security tools

By knowing that the system call table loses integrity, we can infer that something is wrong!
Representative System Overview

- Integrity-based defense system
  - Derive the specifications for certain integrity properties: e.g., data invariants
  - Retrofit monitors or guards to improve the runtime integrity
Data Invariant Based Detection

[Computers & Security, Vol. 43, June 2014]

• Data invariants
  – Constant invariant (e.g., \( b = 10 \))
  – Membership invariant (e.g., \( a \in \{0, 2, 3\} \))
  – Bound invariant, e.g., \((a \geq 0) \land (a \leq 3)\)
  – Non-zero invariant (e.g., \( b \neq 0 \))

How to evaluate the effectiveness of the detection, given that points-to analysis is undecidable?
• False positives
• False negatives
Test Cases Used in Experimental Evaluation

- Benign test cases for Linux

<table>
<thead>
<tr>
<th>Test program</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ltp</td>
<td>Linux Test Project: more than 700 test cases for the Linux kernel and more than 60 test cases for the network stack</td>
</tr>
<tr>
<td>Iperf</td>
<td>A network testing tool that measures the throughput of a network, thus exercising the network subsystem of the kernel</td>
</tr>
<tr>
<td>Andrew benchmark</td>
<td>A file system benchmark</td>
</tr>
<tr>
<td>Miscellaneous</td>
<td>Kernel compilation, ssh, scp, common commands</td>
</tr>
</tbody>
</table>

- Malicious test cases: real-world and synthetic rootkits
Experimental Evaluation of Accuracy

- False positive: one out of 141,280 (Linux), one out of 100,822 (WRK)
- False negative: successfully detect 10 real-world rootkit for Linux, 9 real-world and one synthetic kernel malware for WRK

<table>
<thead>
<tr>
<th>Kernel</th>
<th>Name</th>
<th>Violated Invariants</th>
</tr>
</thead>
<tbody>
<tr>
<td>Linux kernel</td>
<td>Adore 0.42</td>
<td>sys_call_table[2,4,5,6,18,37,39,84,106,107,120,141,195,196,220]</td>
</tr>
<tr>
<td></td>
<td>Adore 0.53</td>
<td>sys_call_table[1,2,6,26,37,39,120,141,220]</td>
</tr>
<tr>
<td></td>
<td>All-root</td>
<td>sys_call_table[24]</td>
</tr>
<tr>
<td></td>
<td>Kbdv2</td>
<td>sys_call_table[24,106]</td>
</tr>
<tr>
<td></td>
<td>Kbdv3</td>
<td>sys_call_table[30,199]</td>
</tr>
<tr>
<td></td>
<td>Modhide</td>
<td>sys_call_table[5]</td>
</tr>
<tr>
<td></td>
<td>Phide</td>
<td>sys_call_table[2,37,141]</td>
</tr>
<tr>
<td></td>
<td>Rial</td>
<td>sys_call_table[3,5,6,141,167]</td>
</tr>
<tr>
<td></td>
<td>Rkit 1.01</td>
<td>sys_call_table[23]</td>
</tr>
<tr>
<td></td>
<td>Suckit 2</td>
<td>sys_call_table[59]</td>
</tr>
<tr>
<td>WRK</td>
<td>Real-world Rootkits</td>
<td>KiServiceTable[185]</td>
</tr>
<tr>
<td></td>
<td>Alureon</td>
<td>KiServiceTable[x], where x=68,75,77,126,256</td>
</tr>
<tr>
<td></td>
<td>Bot Mailer 2</td>
<td>IDT[0], IDT[1], ..., IDT[255]</td>
</tr>
<tr>
<td></td>
<td>Cutwail</td>
<td>KiServiceTable[x], where x=49,50,128,134,151,181</td>
</tr>
<tr>
<td></td>
<td>Haxdoor</td>
<td>KiServiceTable[x], where x=49,50,128,134,151,181</td>
</tr>
<tr>
<td></td>
<td>Rustock.A</td>
<td>IDT[0], IDT[1], ..., IDT[255]</td>
</tr>
<tr>
<td></td>
<td>Rustock.B</td>
<td>IDT[0], IDT[1], ..., IDT[255]</td>
</tr>
<tr>
<td></td>
<td>Storm</td>
<td>KiServiceTable[77], KiServiceTable[151]</td>
</tr>
<tr>
<td></td>
<td>TDL</td>
<td>KiServiceTable[185]</td>
</tr>
<tr>
<td></td>
<td>Trojan.Mssync</td>
<td>KiServiceTable[x], where x=39,43,75,77,122,125,151,181</td>
</tr>
<tr>
<td>Proof-of-Concept Malware Sample</td>
<td>Crash Kernel</td>
<td>0 ≤ ExpPoolScanCount ≤ 31</td>
</tr>
</tbody>
</table>
An Example Large-Scale Malware Behavior Analysis Experiment

- Utilize a virtualized environment (VMWare) to minimize the risk of malware propagation and ease the experimental environment setup (VM snapshots)
- Utilize the Cuckoo sandbox to capture malware’s behavior (system calls, files, etc)
- Utilize a home-grown tool to monitor malware’s kernel-level activities
- 63,200 malware samples automatically executed and analyzed
- Took weeks to finish
Virtualized Threat Monitoring
Automation T&E System Platform

- Focus is on the test and evaluation (T&E) technology that is capable of planning, deploying, monitoring, execution, visualization & automation of threats in virtualized environment
My Perspective

• The role of experimental science and research infrastructure in the cybersecurity space
  – To evaluate a solution
    • Metrics hard to precisely quantify, e.g., this solution improves the security, by how much?
  – To verify a hypothesis
    • E.g., malware often uses mutex as infection markers
  – To understand a threat yet unknown
Future Infrastructure Needs

• Automated planning and deployment tools
• Workload generators (at given rate, pattern, etc)
• Monitoring (measurement) tools
  – Whole-system emulators
  – Instrumentation tools (custom features not provided by the original program)
  – Network monitoring tools (such as Wireshark)
• Experiment control tools
  – start/stop/abort/pause/resume/update
• Result analysis tools
  – Log parsing, data analysis, taint tracking, threat ontology
Thank You!

Jinpeng Wei
Email: weijp@cs.fiu.edu
http://www.cs.fiu.edu/~weijp
Research Topic: Result Integrity of MapReduce Computation on Public Cloud

Diagram:
- First Layer Check
- Second Layer Check
- Task Queue: ... t3 t2 t1
- Public Cloud
- History Cache of VM1
- W1
- R2 = R2'
- Verify?
- No
- Yes
- Worker W1 is benign
- Accept W1 results; Empty History Cache of W1
- No
- Accept Result In Batch
- Job Error Rate vs Cheat Probability
  - T = 50, v = 0.15
  - T = 600, v = 0.07, r = 0.16
  - m= 0.167
  - m= 0.5
  - m= 1