Putting Trust in Malicious Systems

Jon Giffin
Georgia Tech

giffin@cc.gatech.edu
Defense: can security software reliably use virtual environments?

Design: what will hypervisors look like in five years?
Traditional trust

Truisms: Whoever controls the lowest layer wins

Traditional software security architectures rely on unsafe trust relationship
Kernel-level malware

Unhook process from accounting list
Kernel-level malware

Unhook process from accounting list
Modern architecture

Security VM
- Trusted applications
  - Security software
- Trusted OS
  - Security driver

Guest VM
- Applications
  - Malware
- OS
  - Malware

Shared memory

Hypervisor

Hardware
**Virtual machine introspection**

- Assign semantic meaning to raw memory bytes
  - Exported symbols
  - Predefined knowledge of kernel data types

---

**Diagram:**

```
Process: task_struct ----> task_struct
Linked list iteration
```

List of running processes
Virtual machine introspection

So... attackers can alter kernel dynamic data structures...

... and VMI-based utilities will build the incorrect view of victim VM...

... in the same way they would have had they been executing directly on the victim machine.
Trust inversion

Security VM

- Trusted applications
  - Security software
- Trusted OS
  - Security driver

Guest VM

- Applications
  - Malware
- OS
  - Malware

Inverted trust

Hypervisor

Hardware
Memory protection

Kernel memory:

Process memory:
Memory protection

Kernel memory:

- Kernel code: Writes allowed
- Kernel data: Writes allowed
- Driver code: Writes prevented
- Driver data: Writes allowed
Kernel memory access control

Kernel memory:

- Hypervisor & security VM enforce access control policy on kernel memory
- Policy: security-critical kernel data can be written only by core (non-driver) kernel code
Policy enforcement

- How is provenance of write determined?
  - Stackwalk (pray for frame pointers)
  - Other techniques for corner cases
Policy enforcement

• Reducing performance cost
  – Kernel data structure layout optimizations
  – Kernel memory allocation optimizations

Make this:

Look like this:
Kernel memory access control

Security VM

- Trusted applications
  - Security software
- Trusted OS
  - Security driver

Guest VM

- Applications
  - Malware
- OS
  - Malware

Inverted trust

Hypervisor

Hardware
Conclusions

Trust assumptions made by VMI-based software require consideration

Hypervisors becoming kernels...
kernels becoming libraries

Kernel-style memory protection restores trust in guest memory views
Questions?

Jon Giffin
Georgia Tech

giffin@cc.gatech.edu