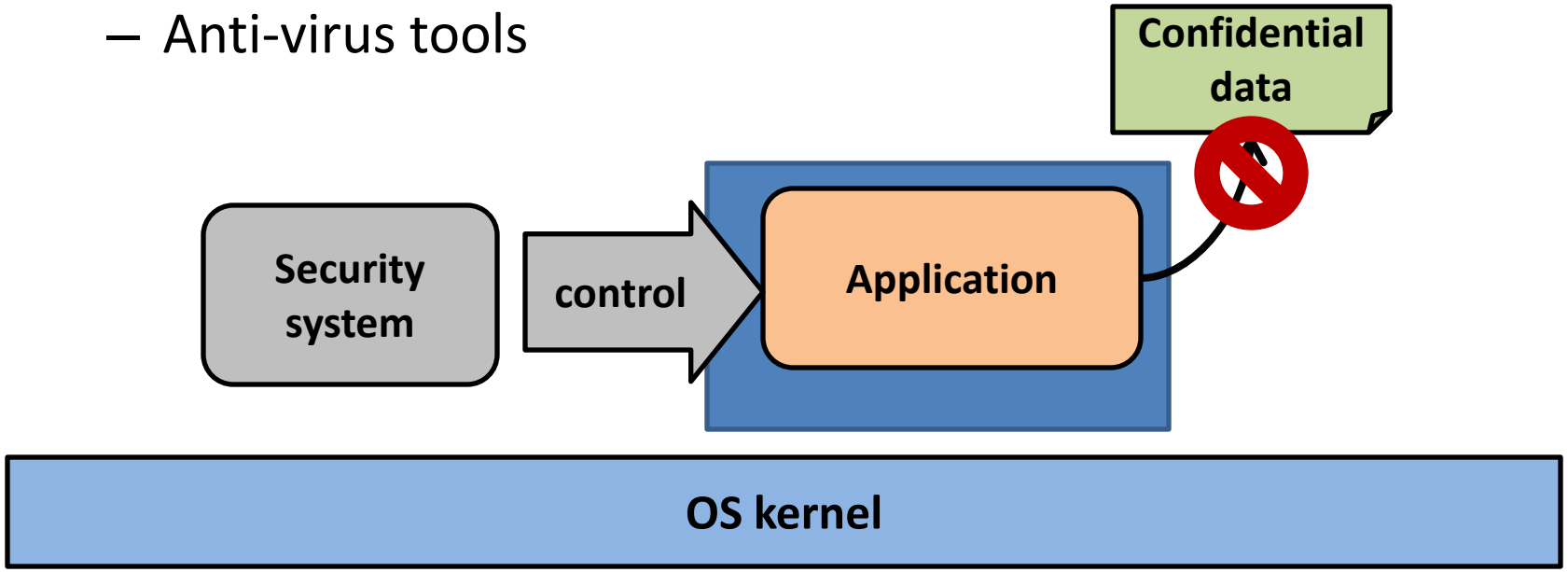
Controlling System Calls and Protecting Application Data in Virtual Machines

Koichi Onoue* Yoshihiro Oyama** Akinori Yonezawa*

* The University of Tokyo
** The University of Electro-Communications

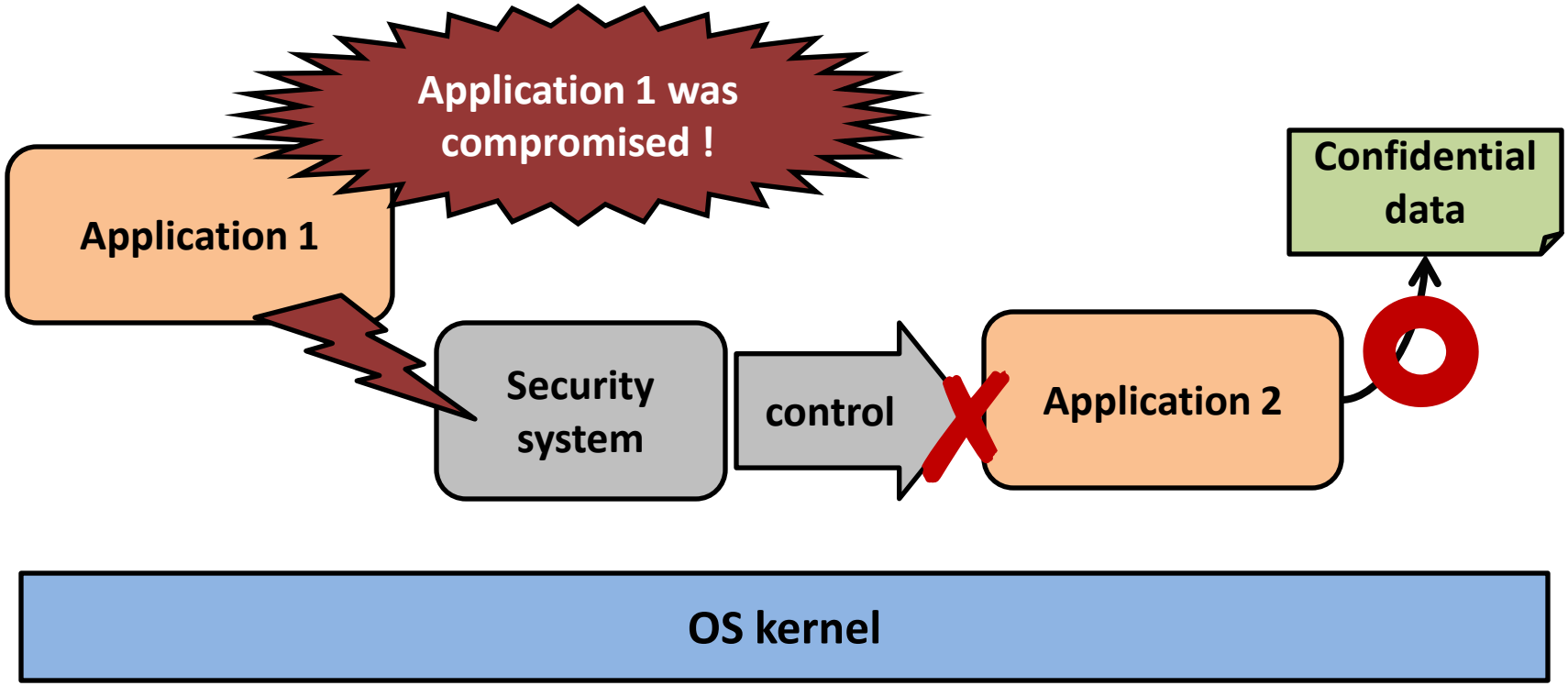
Protection for Applications

- Security systems has been widely applied to provide secure computing environments
  - Sandboxing systems
  - Intrusion detecion/prevension systems (IDSes/IPSes)
  - Anti-virus tools



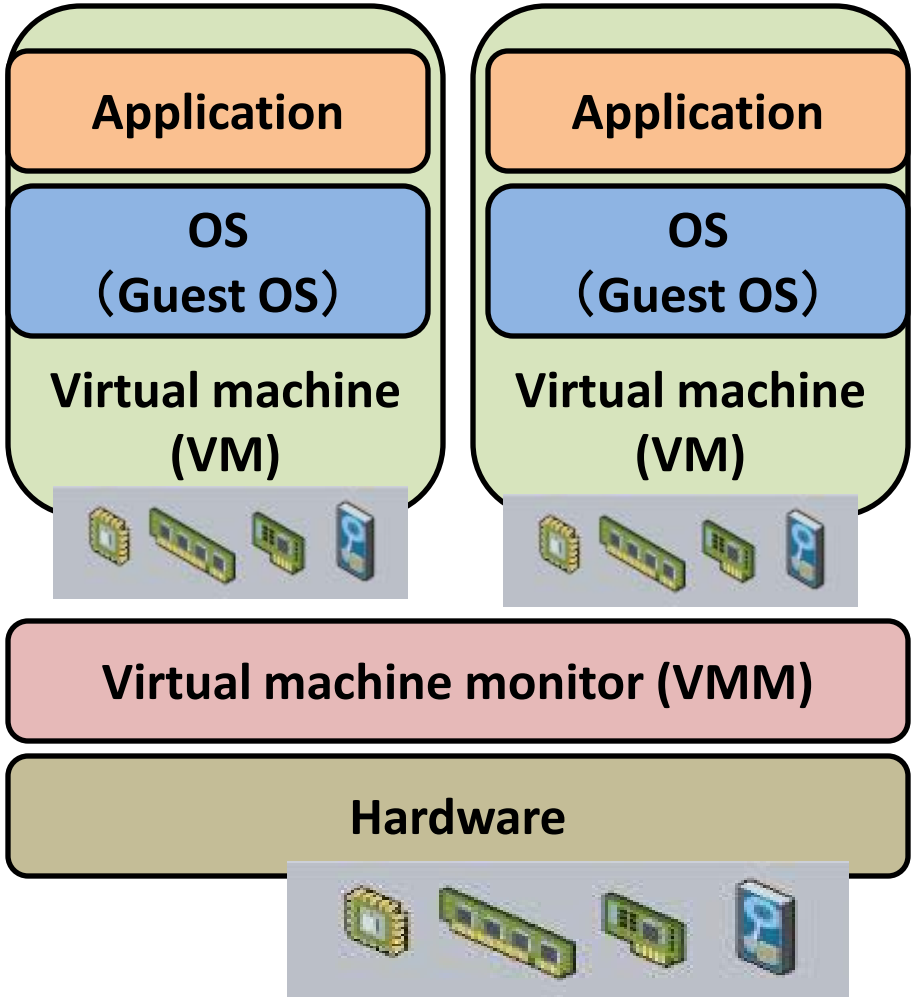
Security Systems Can Also Be Compromised !

- Security systems and the other applications are running in the same execution space



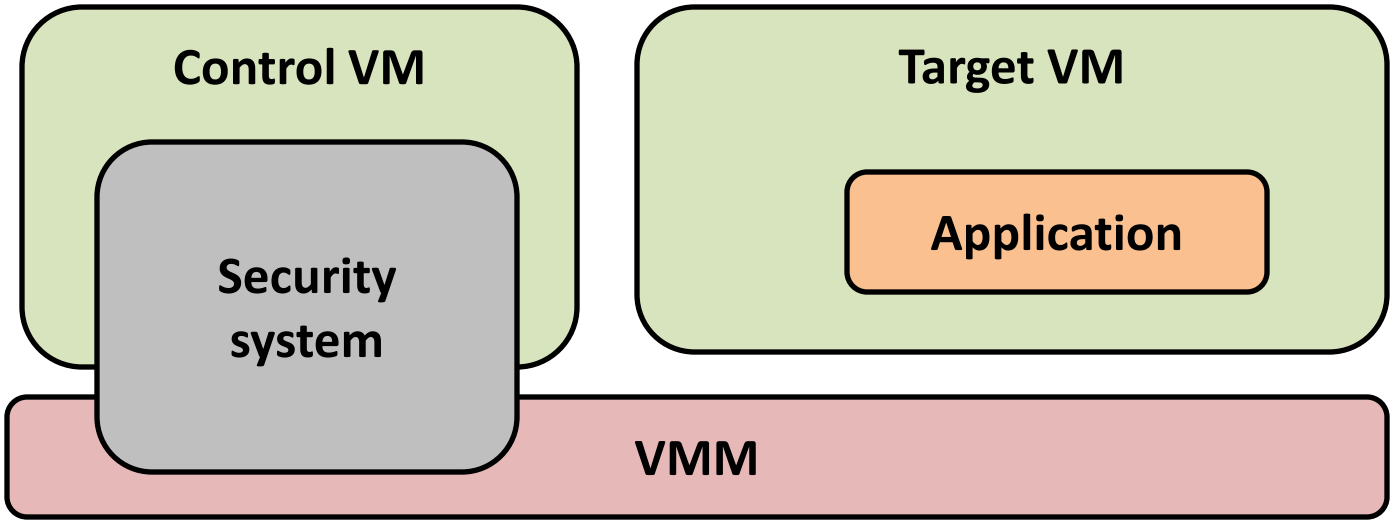
Advantages of Virtual Machine Monitor (VMM) in Terms of Security

- VMM provides strong isolation between VMs
  - VMM prevents a compromised VM from attacking the other VMs
- VMM can control access to physical resources such as physical memory and a disk
  - VMM is running at the higher privileged level than VMs



Our Goal

- Enhancing application security by a system running outside of VMs
  - In cooperation with VMM, the security system controls behaviors of application and protects application data



Our Approach

- Our system consists of program in VMM and program in control VM
  - They run outside of target VMs
- It controls system calls invoked by application process
- It controls memory and file operations related to target applications

✓ Our system controls only the target applications specified by users

We extend a para-virtualization version of Xen

Controlling System Calls from Outside of target VMs

Comparison between “w/o VMM” and “w/ VMM”

	Security systems (“w/o VMM”)	**Security system in cooperation with VMM (“w/ VMM”)**
Attack against security systems	✗ Not hard	○ Hard
Execution states obtained by security systems	○ OS-level	✗ Hardware-level

Goal for Controlling System Calls

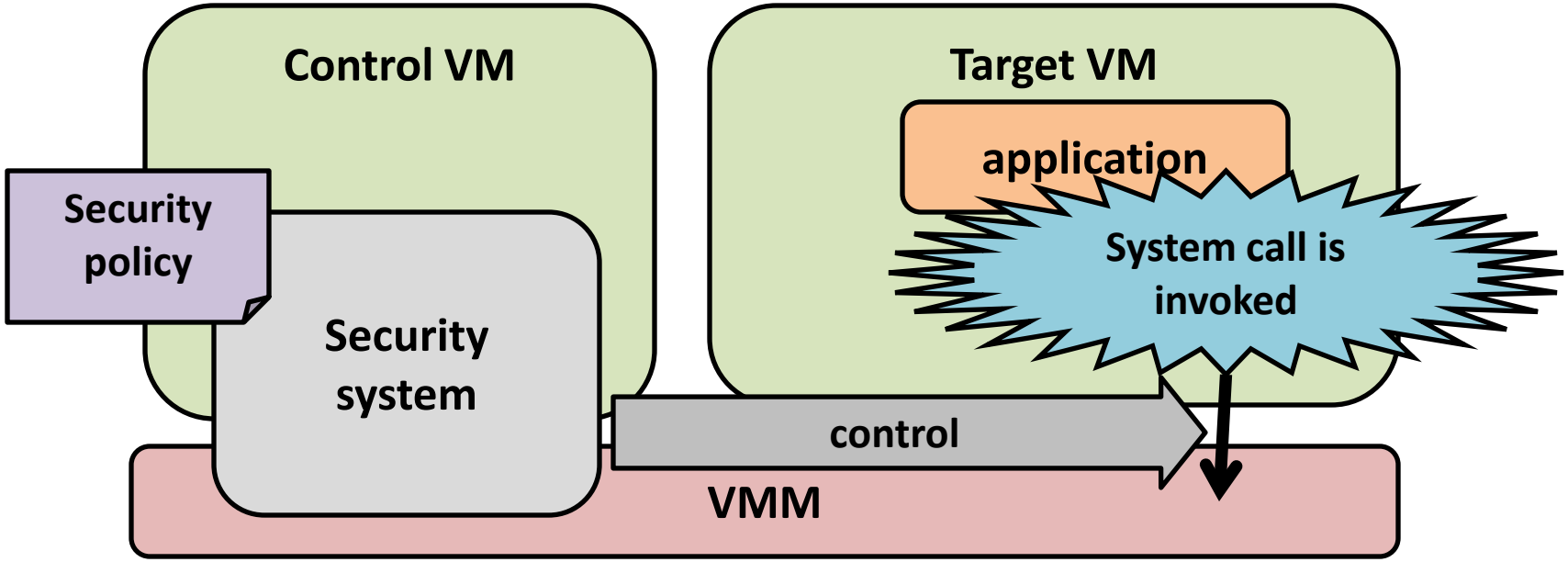
	Security systems	**Security systems in cooperation with VMM**
Attack against security systems	✗ Not hard	○ Hard
Execution states obtained by security systems	○ OS-level	✗ Hardware-level

Semantic gap

Our goal

Approach to Controlling System Calls

- Controlling system calls from outside of VMs
  - Using information on target OSes created in kernel build
  - Conforming to security policies



Bridging the Semantic Gap

- What a VMM can observe
  - Events : Privileged instructions, interrupts, …
  - Execution States : Registers, memory pages, …

- What security systems require
  - Events : System calls, …
  - Execution states : Process ID, system call number, …

Security Policy

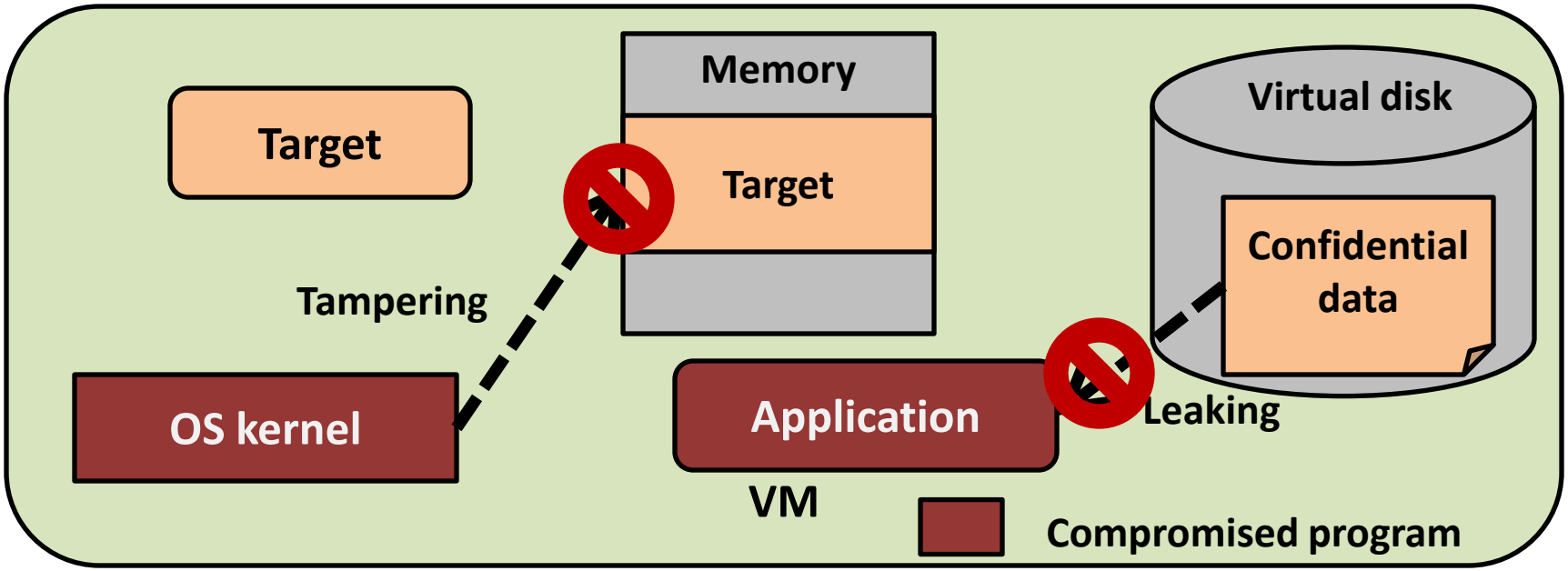
- Specifies controlled system calls with pattern matching

```
…
open  default:  allow
    fileEq(“/etc/passwd”)
    or filePrefixEq(“/etc/cron.d”)
        deny(EPERM)
…
```

Controlling Memory and File Operations Related to Application Data

Goal for Protecting Application Data

- Prevent compromised programs from leaking target data and tampering with them
  - We assume attackers read/write application data with ptrace system call and kernel modules, etc.



Approach to Protecting Application Data

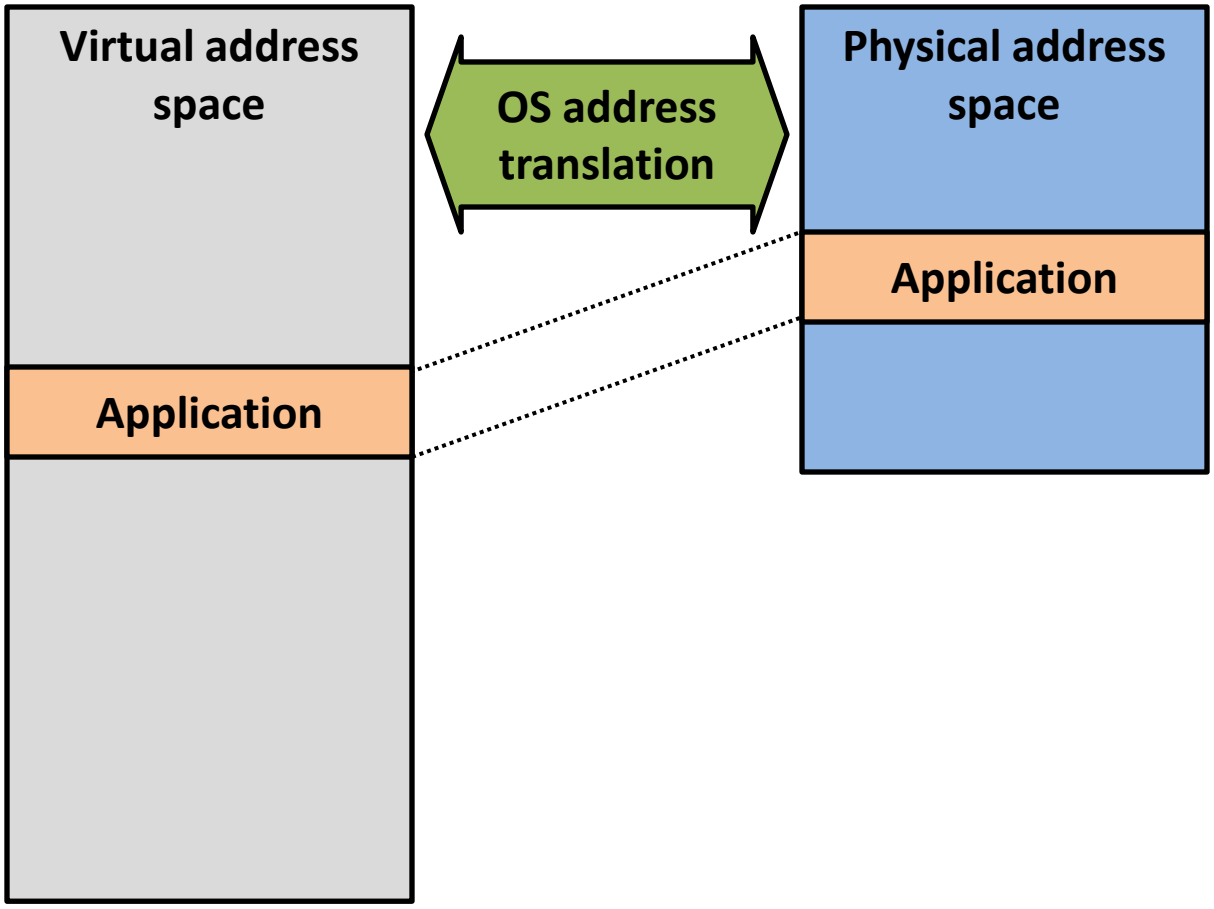
- Hiding “real” application data on memory and a virtual disk from compromised programs
  - Compromised programs include target OS kernel

- Application data on memory
  - Code region, data region, stack region, etc.

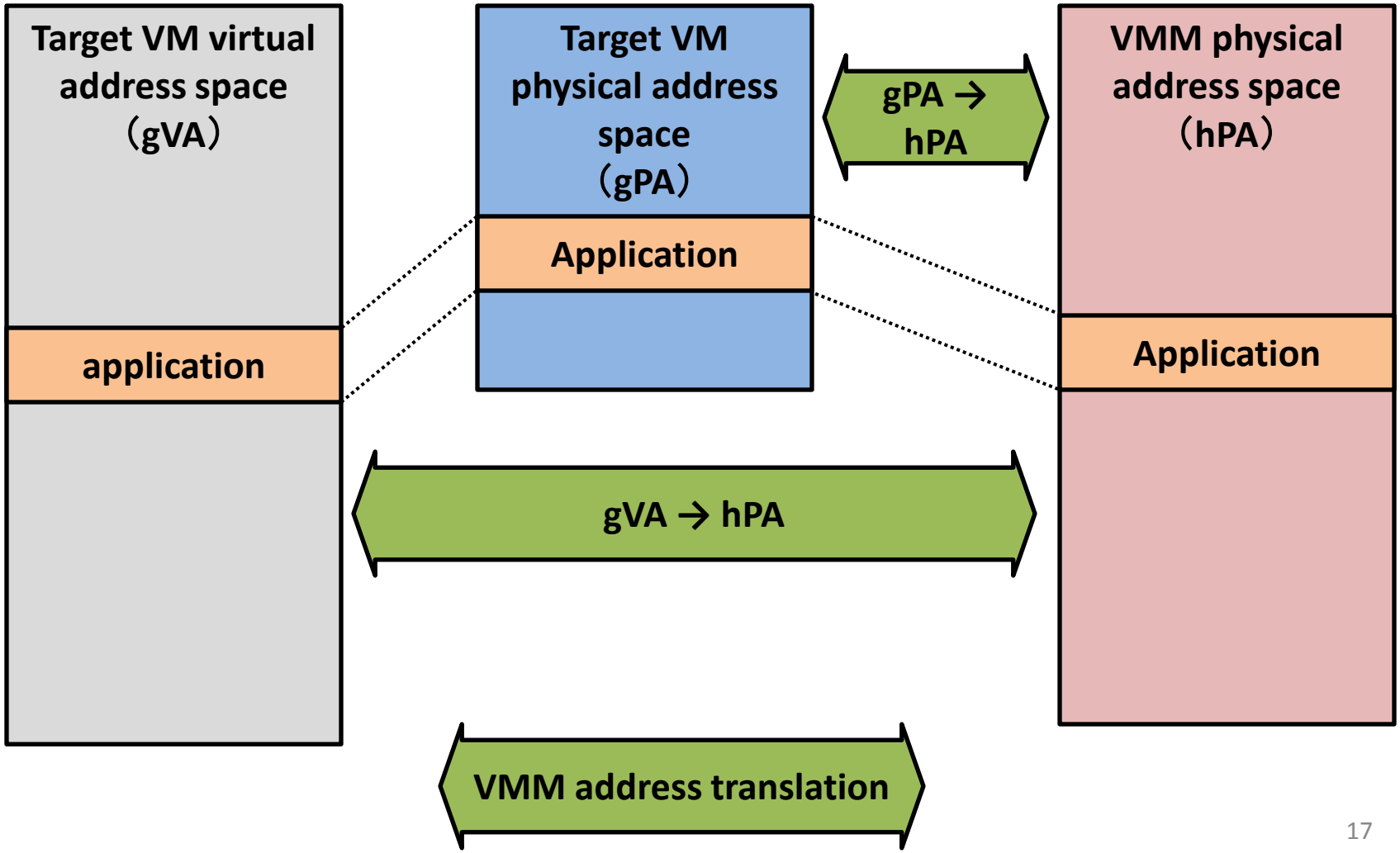
  ➔VMM multiplexes physical pages
    - Overshadow[Chen et al., 2008]
    - [Rosenblum et al.,2008]
- Application data on a virtual disk
  - Executables, configuration files, etc.

  ➔ Control VM manages them

OS Memory Management

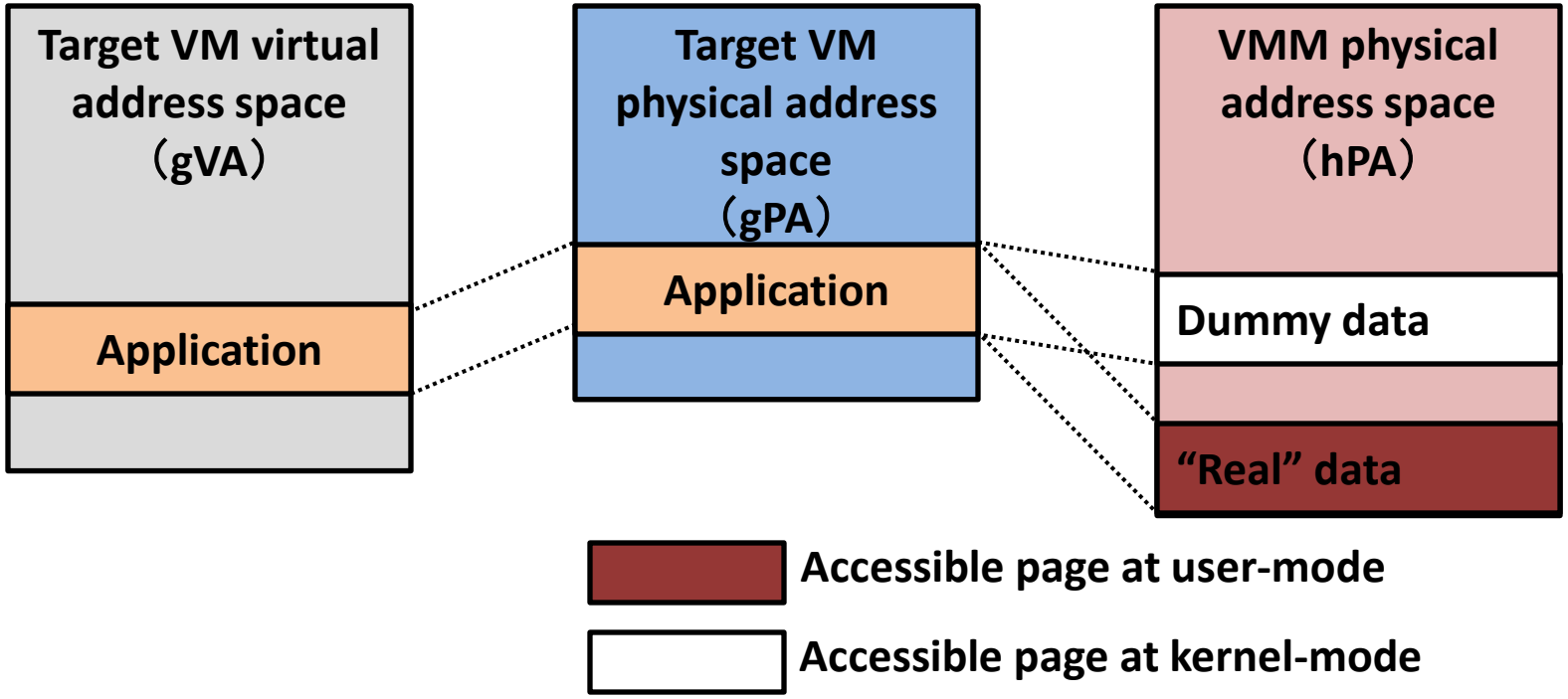


VMM Memory Management



Protecting Memory (1/2)

- According to the operational mode,
  a VMM switches accessible physical pages

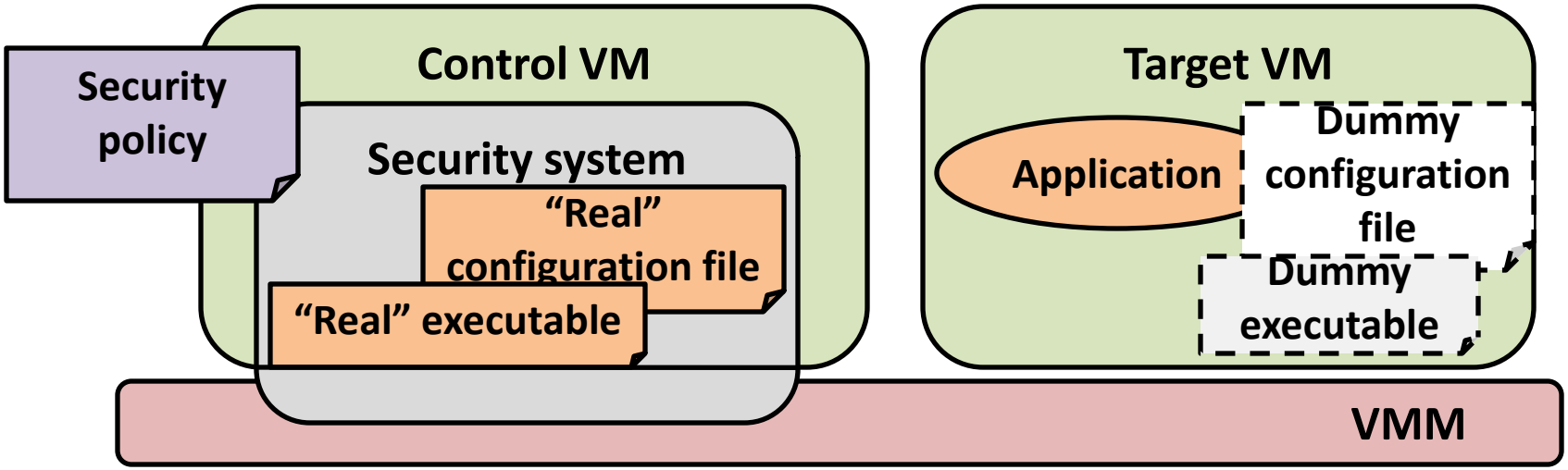


Protecting Memory (2/2)

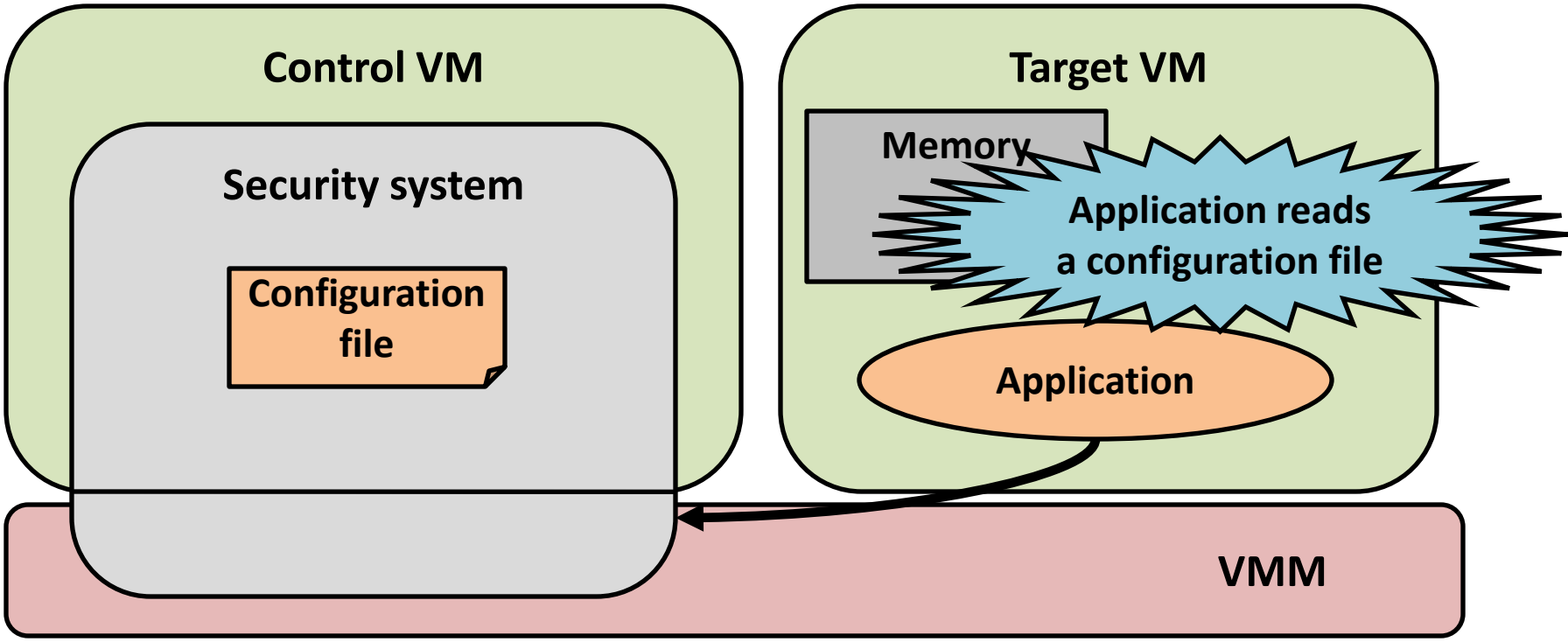
- VMM switches page tables
  when the operational mode is changed
  - Exception/Interrupt handling
  - System call handling

Approach to Protecting Application File Data (1/5)

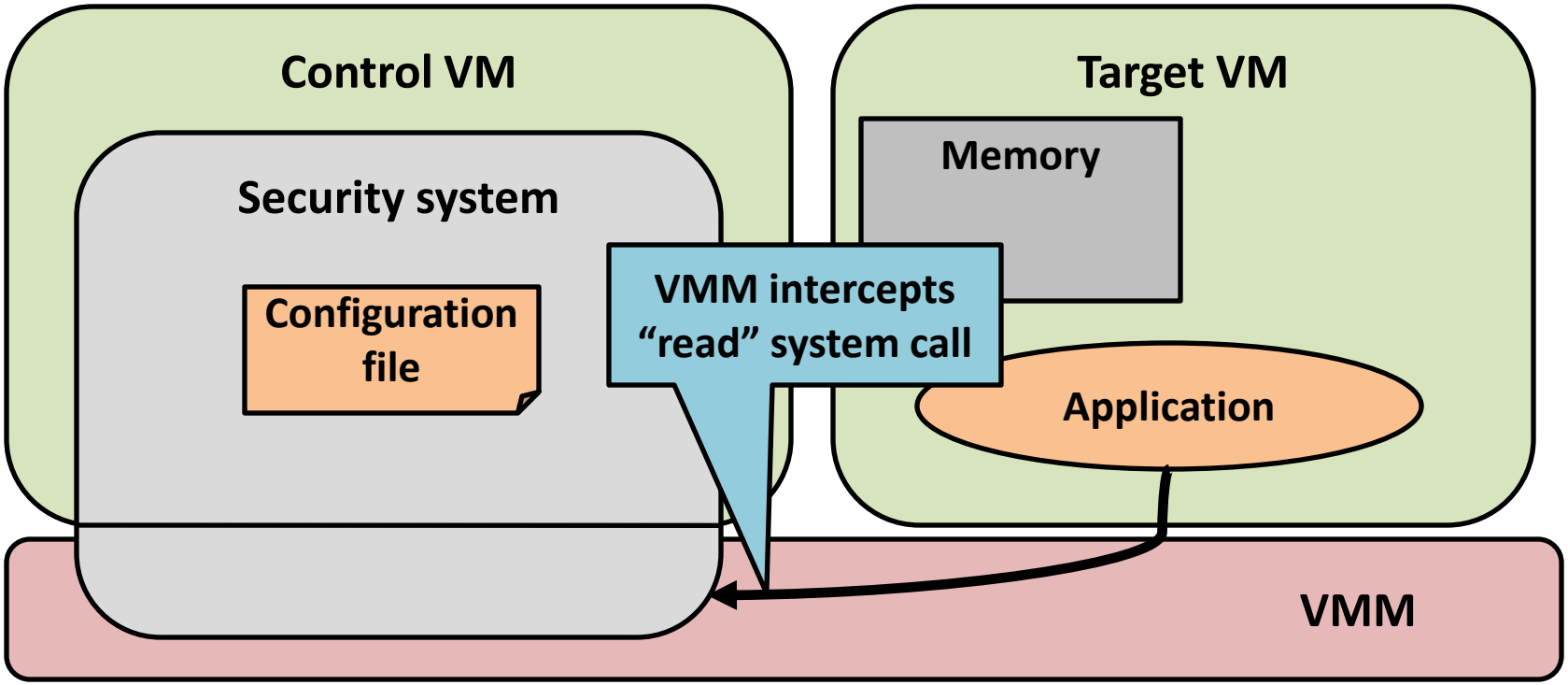
- Control VM manages “real” target files
  - Executables, configuration and data base files, etc.
  - Security policy specifies target files
- Target VM manages “dummy” target files



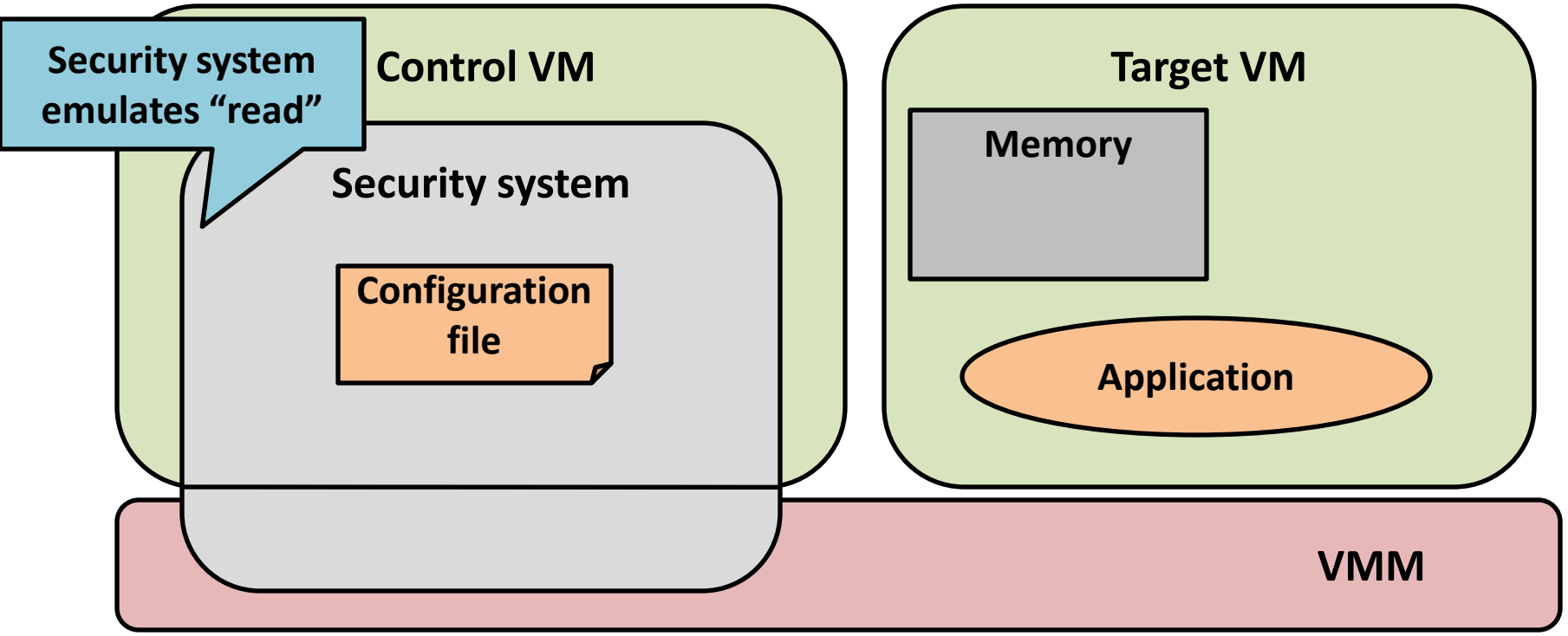
Approach to Protecting Application File Data (2/5)



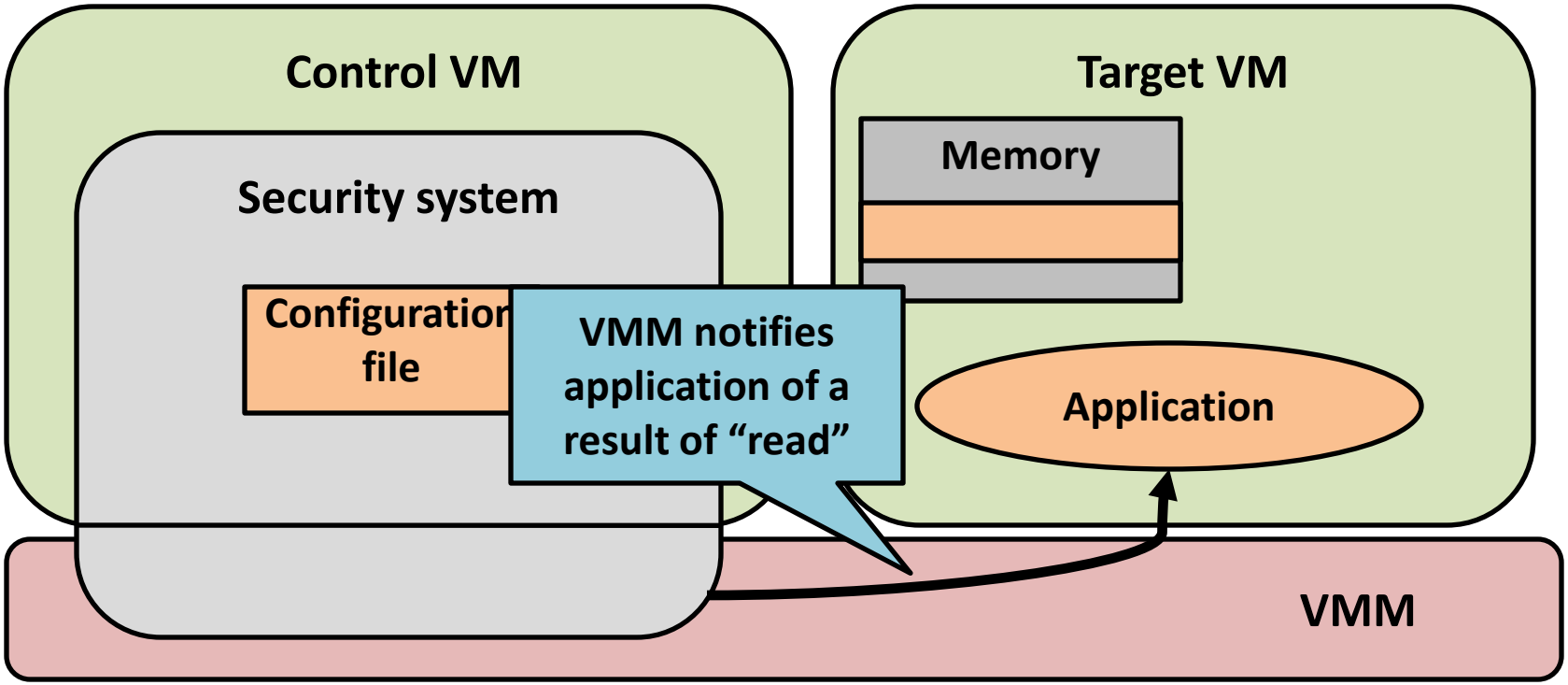
Approach to Protecting Application File Data (3/5)



Approach to Protecting Application File Data (4/5)



Approach to Protecting Application File Data (5/5)



Conclusion

- We have proposed a system that enhances application security inside target VMs
  - Controlling of application behaviors
    - Controlling of system calls from outside of target VMs
  - Protecting application data on memory and a virtual disk
    - Application memory data: VMM multiplexes target physical pages
    - Application file data: Control VM manages them

Thank you for your attention