Examples of Monitored Data

The OS collects different types of data at different layers.

- System call traces: describe the activities or behaviors of processes running in the system.
- Log file: information on user activity, including user’ login record, history of commands, etc.
- File integrity checksums: periodically scan critical files for changes and compare cryptographic checksums for these files, with a record of known good values.
- Registry access: monitor access to the registry. This is specific to Windows operating systems.
- Kernel and driver-level monitoring: this source provides insight into OS kernel-level anomalies.
- Resource usage: CPU, memory or I/O utilization and activities can indicate the execution of some malicious behaviors.
- Network activities: include established connections and received packets

Intrusion Detection

Intrusion Detection System (IDS)

- A system used to detect unauthorized intrusions into computer systems.
- IDS can be implemented at different layers, including network-based IDS, host-based IDS.
- We mainly focus on host-based IDS, which monitors the characteristics of a single host for suspicious activities.

An IDS comprises three logical components:

- Sensors: responsible for collecting data.
- Analyzers: responsible for determining if an intrusion has occurred, and the possible evidence. It may provide guidance about what actions to take as a result of the intrusion.
- User interface: enables a user to view output from the system or control the behavior of the system.

Detection Methodologies

Signature-based detection

- Flag any activity that matches the structure of a known attack
- It is *blacklisting*: keep a list of patterns that are not allowed, and alert if we see something on the list.
- Advantage: simple and easy to build; good at detecting known attacks.
- Disadvantage: cannot catch new attacks without a known signature.

Anomaly-based detection

- Develop a model of what normal activities look like. Alert on any activities that deviates from normal activities.
- It is *whitelisting*: keep a list of allowed patterns, and alert if we see something that is not on the list.
- Advantage: can detect attacks we have not seen before.
- Disadvantage: false positive rate can be high (many non-attacks look unusual).

Outline

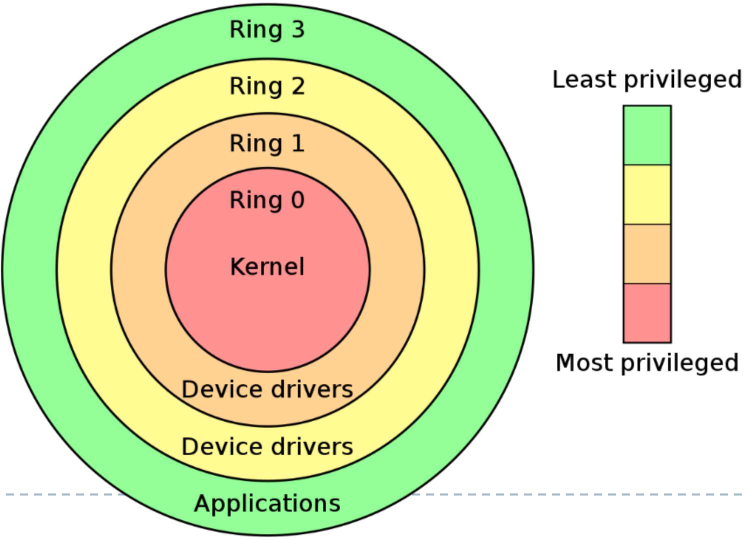
- **Security Protection Stages in OS**
  - Authentication
  - Authorization with Access Control
  - Logging, Monitoring & Auditing

- **Privilege Management in OS**

Privileged Rings Inside OS

Operating modes

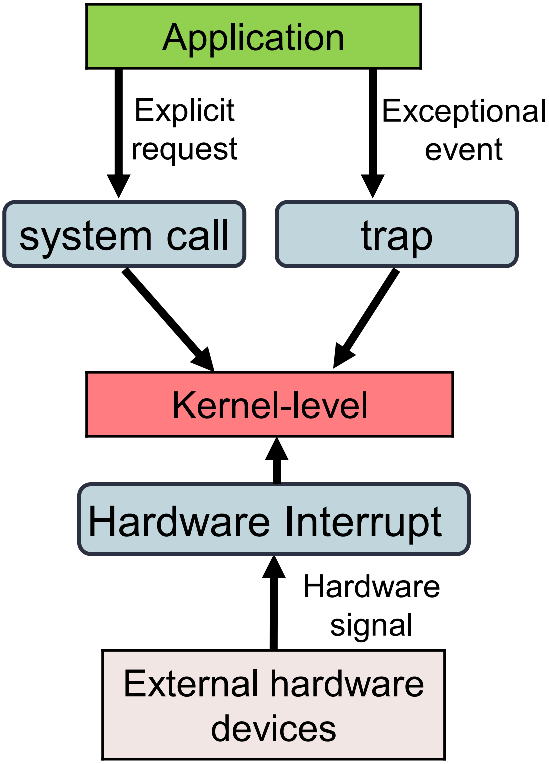
- Kernel mode has the highest privilege, running the critical functions and services; user mode has the least privilege.
- Entities in the higher rings cannot call the functions and access the objects in the lower rings directly.
- Context switch is required to achieve the above procedure, system call, interrupt, etc.
- Status flag allows system to work in different modes.



Context Switch

Different events can trigger the transition from user to kernel levels

- System call: user application explicitly makes a request to kernel for privileged operations
- Trap: user application gets an exceptional event or error and requests the kernel to handle.
- System call and trap belong to software interrupts,
- Hardware interrupt: hardware issues a signal to the CPU to indicate an event needs immediate attention.



Switch procedure

- CPU stores process's states, and switches to the kernel mode by setting the status flag.
- Kernel handles the interrupt based on the interrupt vector in an interrupt table.
- CPU switches back to user mode and restores states

How System Call is Issued and Handled

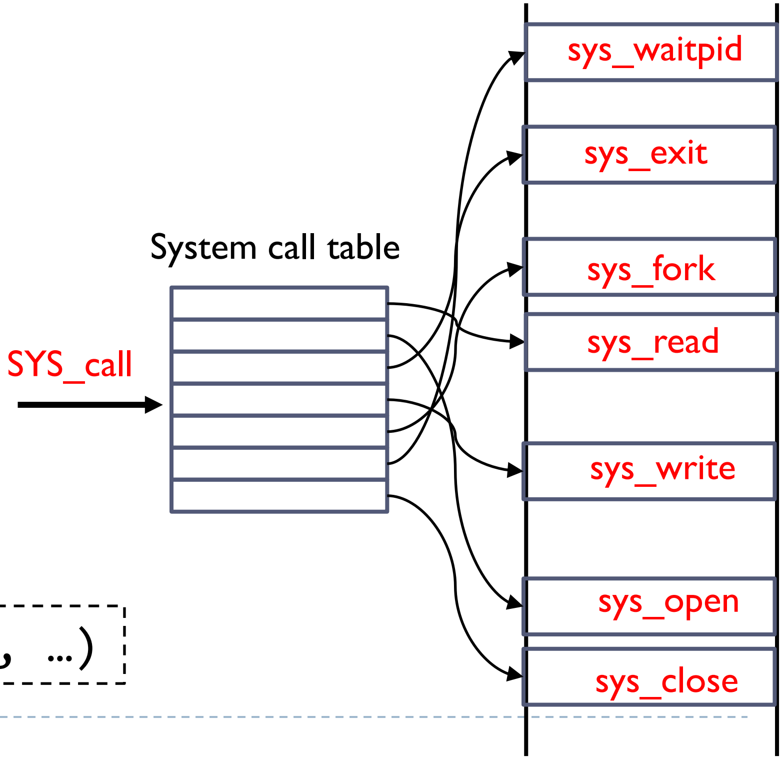
A system call is an interface that allows a user-level process to request functions or services from the kernel level.

- Process control
- File management
- Device management

How to issue a system call?

- System call table: a table of pointers in the kernel region, to different system call functions.
- A user process passes the index of the system call and parameters with the following API:

```
syscall(SYS_call, arg1, arg2, …)
```



Rootkit

Malware that obtains root privileges to compromise the computer

- Root user does not go though any security checks, and can perform any actions to the system
  - Insert and execute arbitrary malicious code in the system's code path
  - Hide its existence, e.g., malicious process, files, network sockets, from being detected.

How can the attacker gain the root privileges?

- Vulnerabilities in the software stack: buffer overflow, format string…

There are some common techniques for rootkits to compromise the systems.