INCHEON NATIONAL UNIVERSITY

CYBERSECURITY CONVERGENCE LAB

Kuberosy+

: Stateful System Call Security Framework for Containers

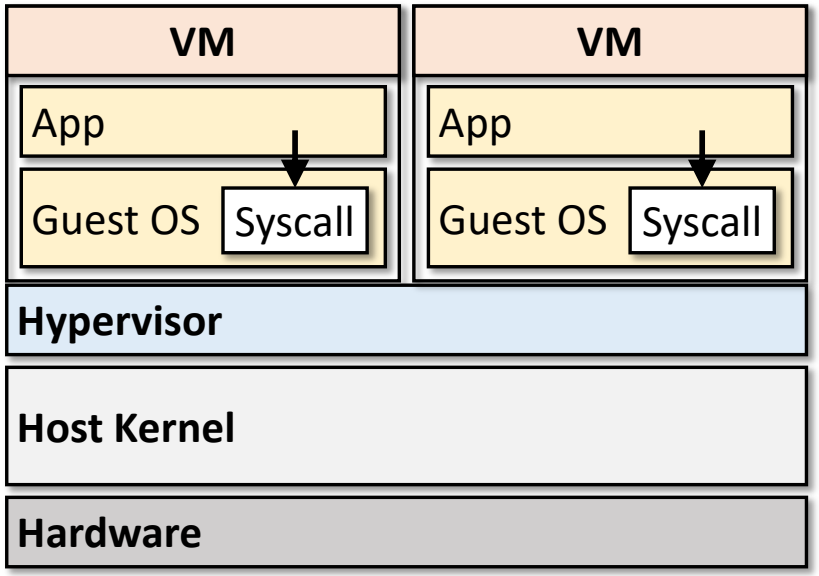
Jin Her

Department of Computer Science & Engineering

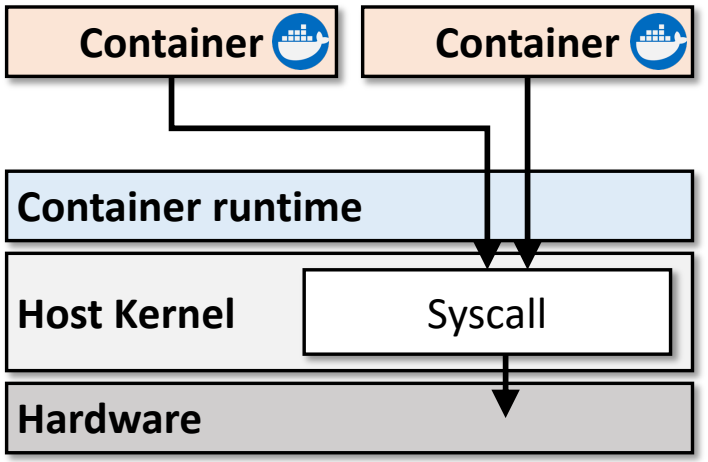
Incheon National University

Background [1] Containerization

- A virtualization approach that has its own guest OS and accesses hardware
- In a containerized environment, host kernel is **shared**



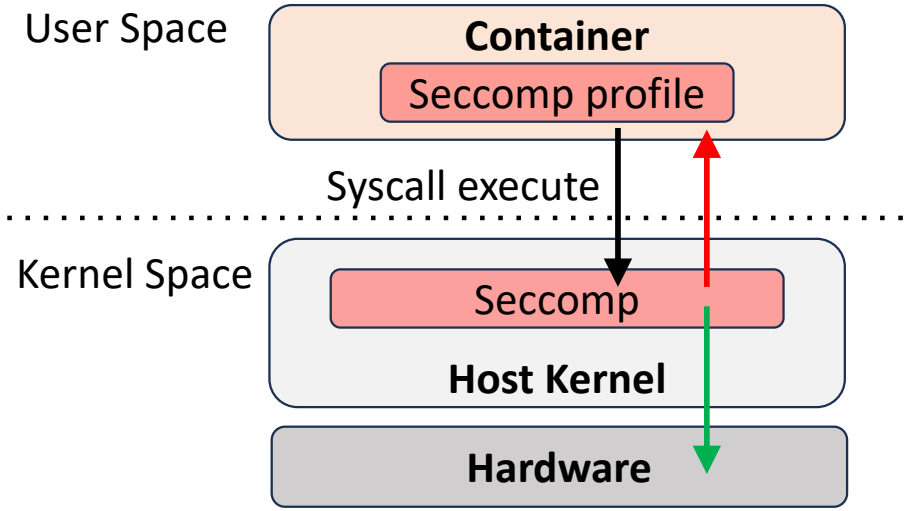
< Virtualization >



< Containerization >

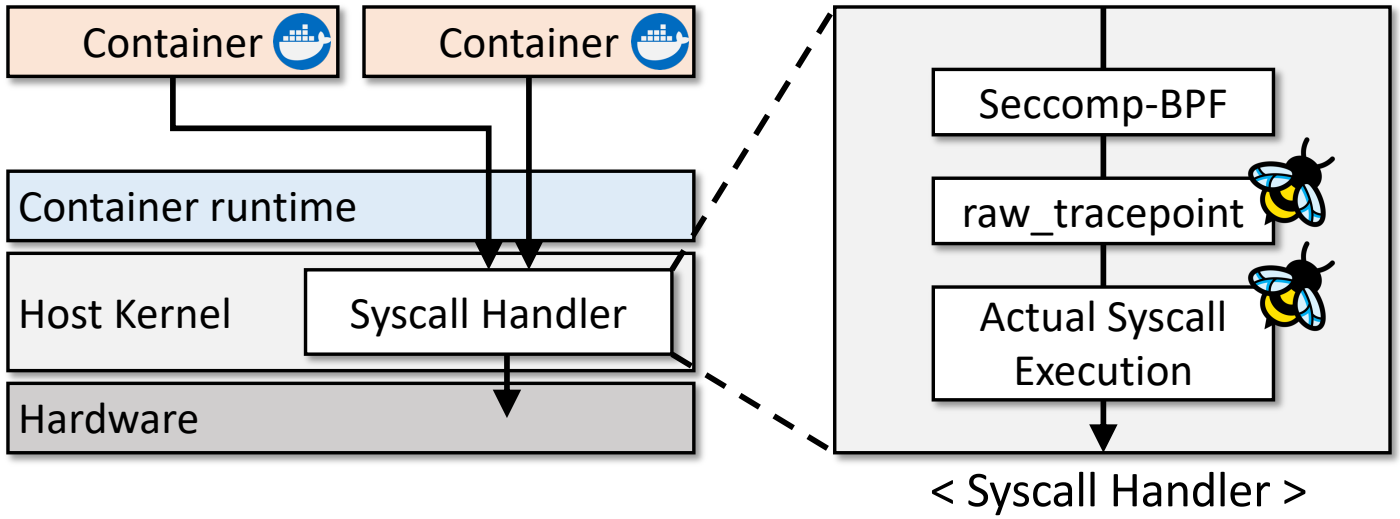
Background [2] Seccomp

- In a container environment, Seccomp cannot change dynamically when the container is running
- In addition, state information is not available, blocking it in the form of a simple Allow list without understanding the context



Background [3] eBPF

- To overcome these Seccomp limitations, eBPF technology is being introduced that can run user programs in the kernel without modifying the kernel
- Representatively, there are Sigkill and LSM BPF-based blocking techniques for eBPF-based system call blocking



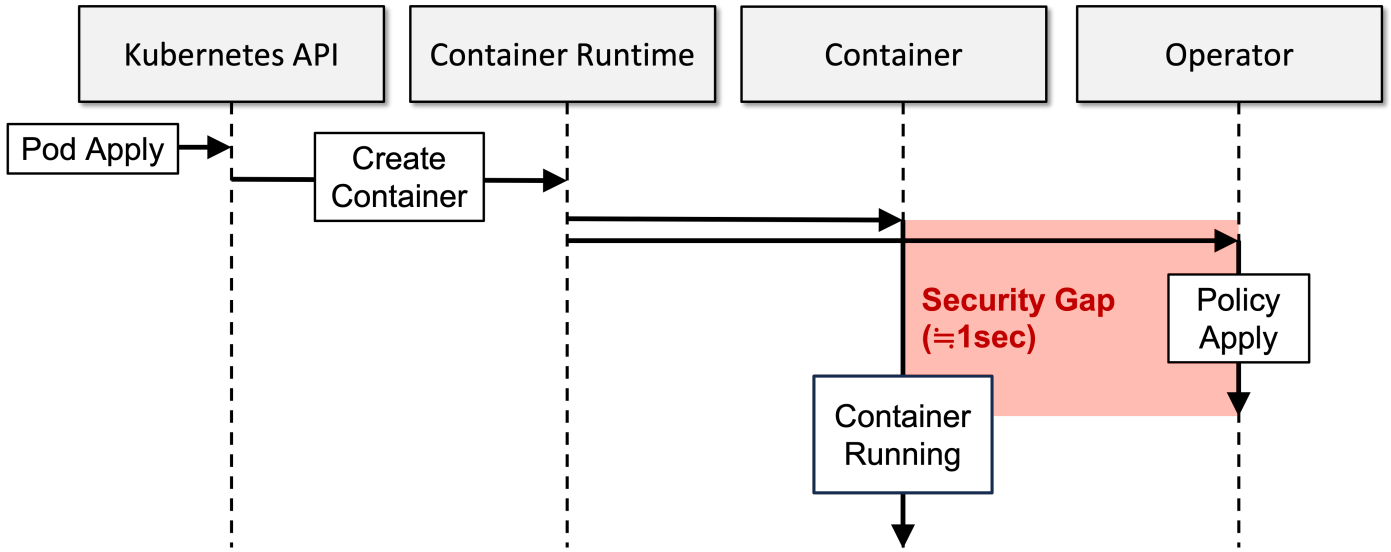
Problem Statement [1] Limitation of Stateless Syscall Filter

- A simple allow list-based system call filter is powerless against malicious, order-based attacks of allowed system calls

	Seccomp-BPF	SIGKILL	LSM-BPF
Dynamic Policy	X	O	O
Stateful Enforcement	X	O	O
Delay-Free Blocking	O	X	O
Full Syscall Coverage	O	O	X

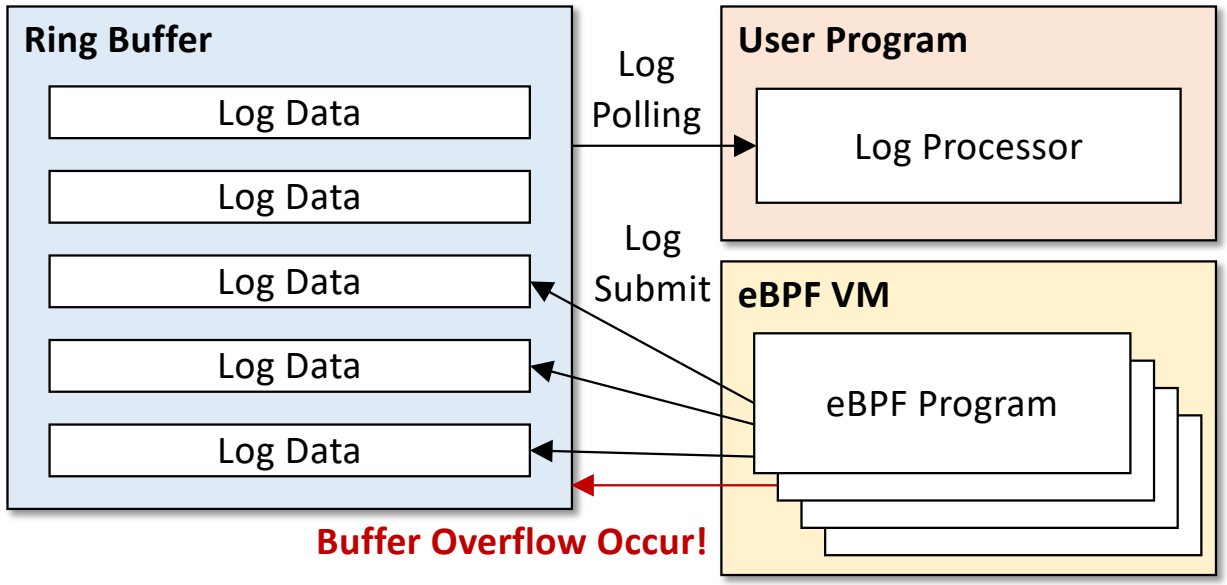
Problem Statement [2] Lack of Lifecycle-Aware Enforcement

- Traditional Operator-based policy application patterns work in parallel with container creation, creating security gaps

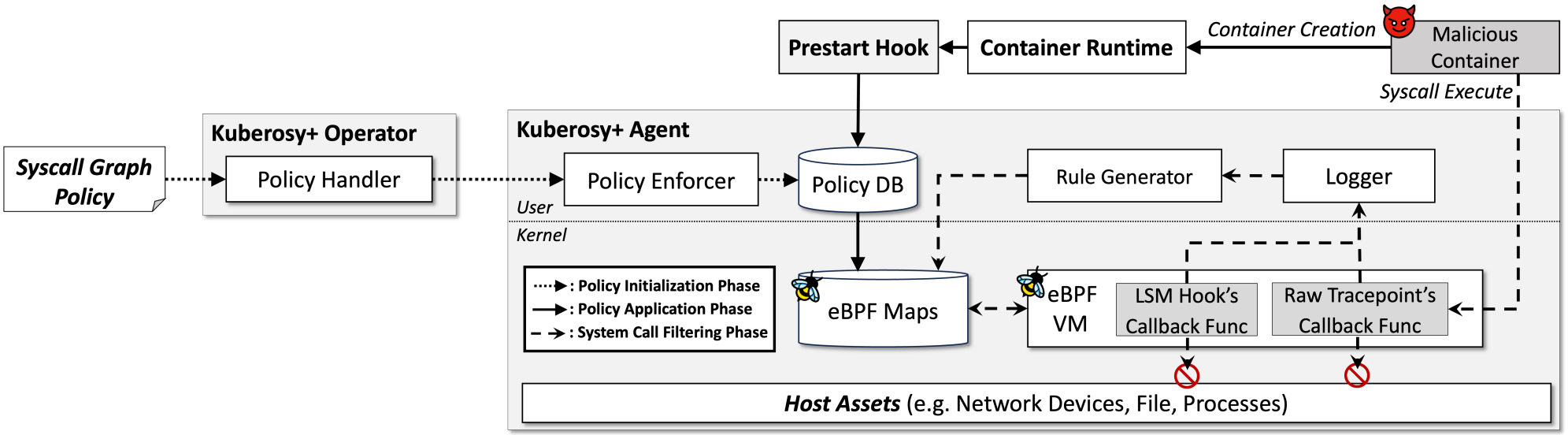


Problem Statement [3] Log Loss in Large Load Environments

- A faster submission rate than log polling in a ring buffer may result in log loss due to buffer overflow

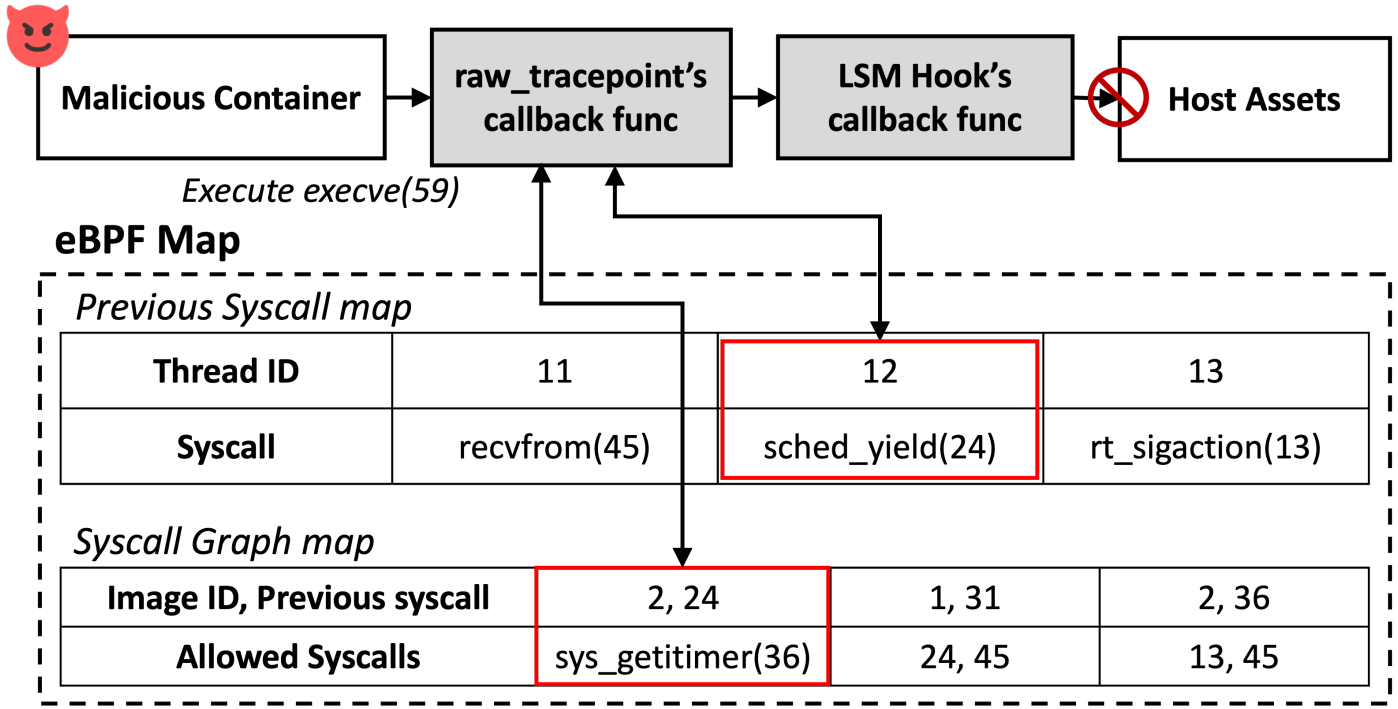


Overall Architecture



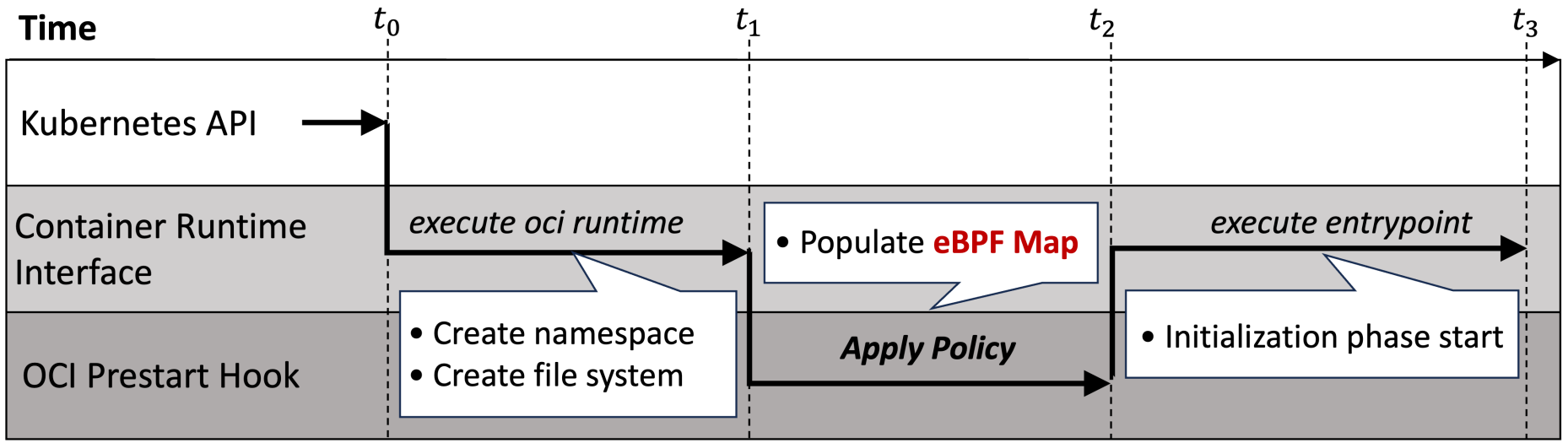
Detail [1] Stateful Syscall Filter

- eBPF Map-based stateful filtering



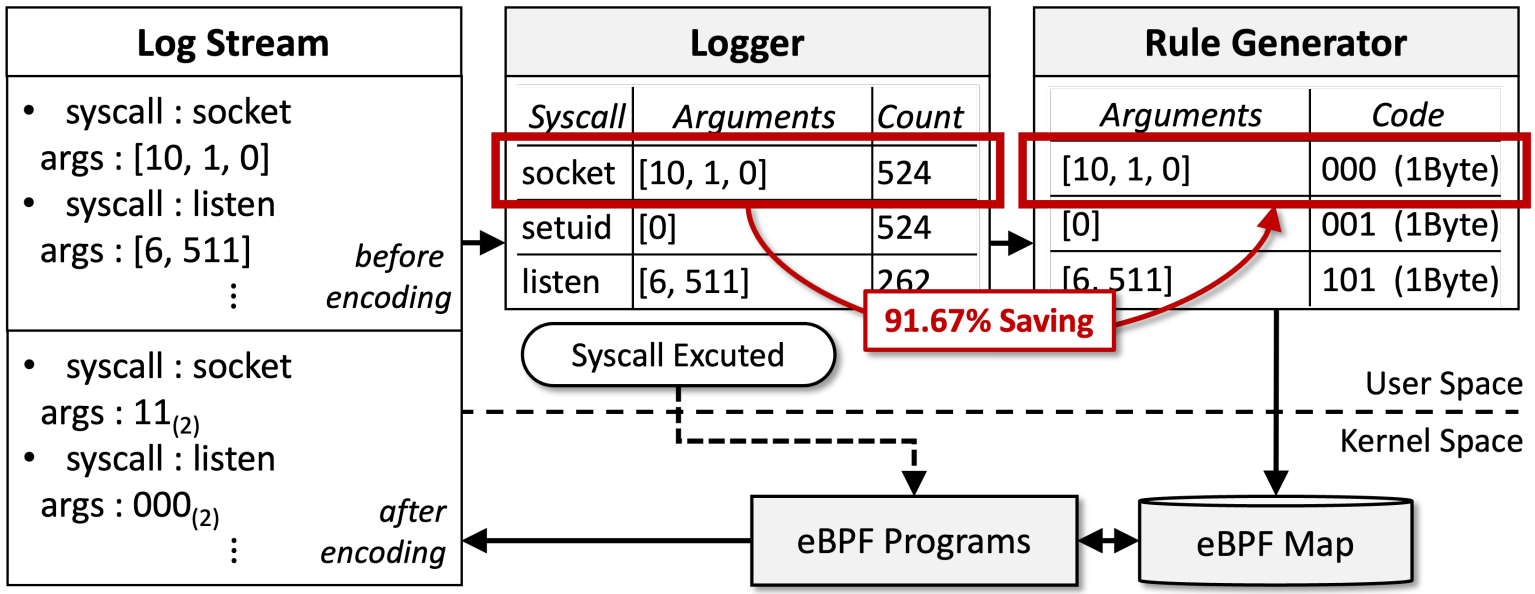
Detail [2] Preemptive Policy Application

- OCI Hook-based preemptive policy application



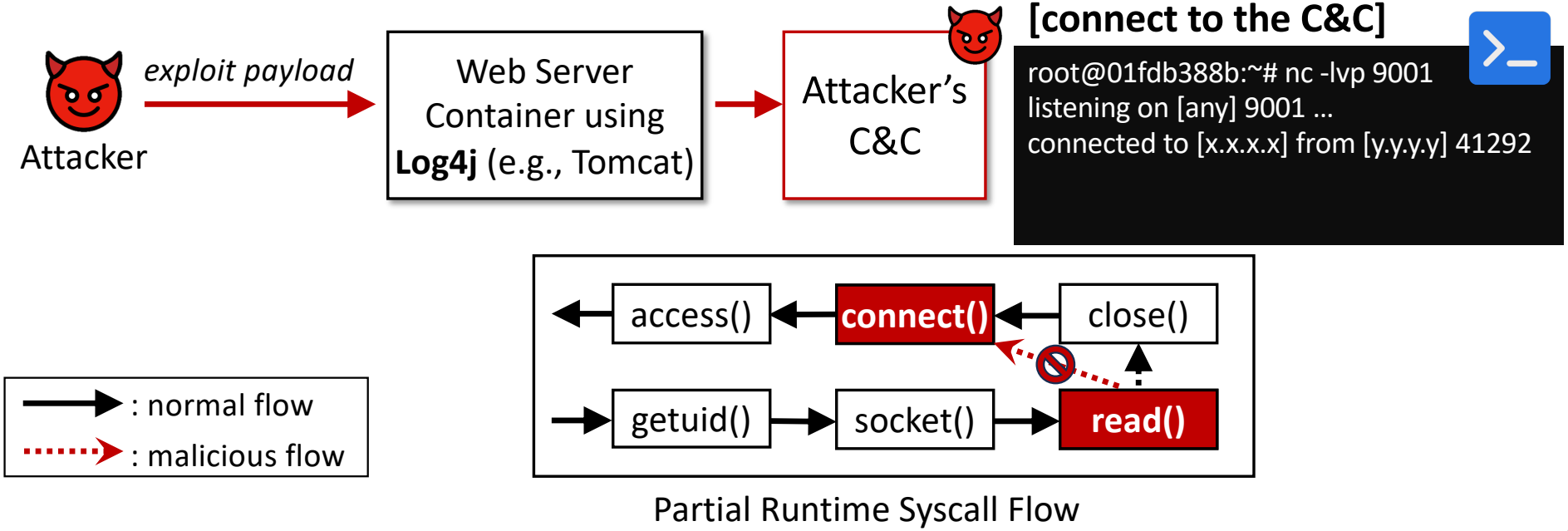
Detail [3] Adaptive Log Compression

- Log compression with high frequency pattern recognition and pattern generation



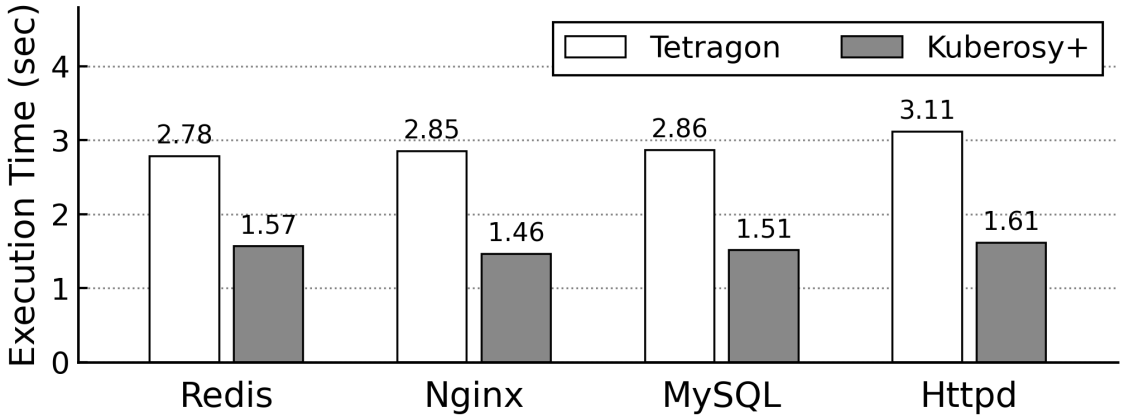
Evaluation [1] Effectiveness of Stateful Enforcement

- Attempt to attack after creating policy based on normal behavior of target
- Attacks are successfully blocked, C&C servers do not receive a response



Evaluation [2] Policy Enforcement Latency

- Measure policy application latency from deployment point to policy application for four applications
- Tetragon experiences an additional delay of 1.36 seconds compared to Kuberosy+



Evaluation [3] Effectveness of Log Compression

- Conducts a system call audit of Google's Online Boutique microservices
- The existing method achieved a compression rate of 36% compared to the original, and the proposed technique achieved a compression rate of 70%

Approach	Total Data Size	Compression Ratio
No Compression	242,187B	-
eAudit	153,660B	36.55%
KubeRosy+	72,089B	70.23%

Conclusion

- **Conclusion**
  - Combining state storage-based system call control, preemptive policy injection via OCI Hook, and adaptive log compression to completely address the initial security gaps in containers and the overhead of high-load environments.

- **Future Work**
  - Combining modern system call analysis studies with this framework, we build an integrated pipeline from automatic generation of container security policies to kernel-level real-time execution.

Q&A