UNIMORE
UNIVERSITÀ DEGLI STUDI DI
MODENA E REGGIO EMILIA

TÉCNICO
LISBOA

DEPARTMENT OF
ENGINEERING
SCIENCE

Investigating Security Incidents with Forensic Snapshots in Kubernetes

Lorena Goldoni and Radostin Stoyanov
Collaboration with Adrian Reber - Senior Principal Software Engineer
Dr. Francesco Faenza, Prof. Claudia Canali, Prof. Wes Armour, Prof. Rodrigo Bruno

Red Hat

Invisible Attacks in Containers

How attackers stay stealthy in Kubernetes?

- Masquerade as legitimate workloads[1, 2]
- Leverage native tools & APIs (kubectl, bash, python)[1, 2]
- Maintain persistence (e.g., DaemonSets, CronJobs)[1, 2]
- Exploit unpatched APIs/runtimes without disruption[3, 4]
- Steal credentials silently[4]

[1] MITRE ATT&CK - Containers Matrix
[2] Microsoft's Threat Matrix for Kubernetes
[3] Kubernetes CVE Feed
[4] https://github.com/cdk-team/CDK

Traditional Forensics in Kubernetes

Challenges with Classic Forensic Assumptions

- Assume static hosts and persistent storage
- Require node-level access and workload disruption
- Focus on disk artefacts, missing memory and runtime state
- Fail with ephemeral containers and rescheduled Pods
- Weak evidence attribution and chain of custody in clusters

Schmid et al. *Limits to the Forensic Analysis of Container Applications in Cloud Environments*. (2025)
Gharaibeh et al. *Don't, Stop, Drop, Pause: Forensics of CONtainer CheckPOINTs (ConPoint)*. (2024)

Modern Container Attacks

Cloud-Native, Runtime-Focused Threats[1,2]

- In-memory malware (no disk artifacts)
- Living-off-the-land binaries in containers (bash, wget, python)
- Exploiting container runtime & kernel interfaces
  - e.g., privilege escalation from Pod to node (CVE-2019-5736, CVE-2022-0847)

Kubernetes Attack Surface[3]

- API-based attacks: RBAC misconfigurations, token reuse, API scraping
- Lateral movement via compromised service accounts
- Sidecar injection and workload mutation at runtime

[1] Cloud Native Security Whitepaper
[2] Microsoft's Threat Matrix for Kubernetes
[3] Microsoft Threat Intelligence, *Understanding the Threat Landscape for Kubernetes and Containerized Assets*

Detection vs Investigation

"Detection without investigation leads to ineffective remediation"

	Detection = Awareness *"Did something bad happen?"*	**Investigation = Understand** *"What happened, how, why, and by whom?"*
Goals:	• Identify suspicious activities • Trigger a response → Real-time	• Reconstruct the incident • Understand attacker actions • Determine impacts → Retrospective
Collected Data:	• Indicators / alerts → Event-based → Signal-driven → Ephemeral data	• Evidence / Artifacts / State snapshots → State-based → Evidence-driven → Durable artifacts
Sources:	• Runtime Alerts • Kubernetes Events • Cloud Audit Logs	• Container forensics • Audit log correlation • Checkpoints (file system & memory)

What is a Forensic Container Snapshot?

From Alerts to Digital Evidence

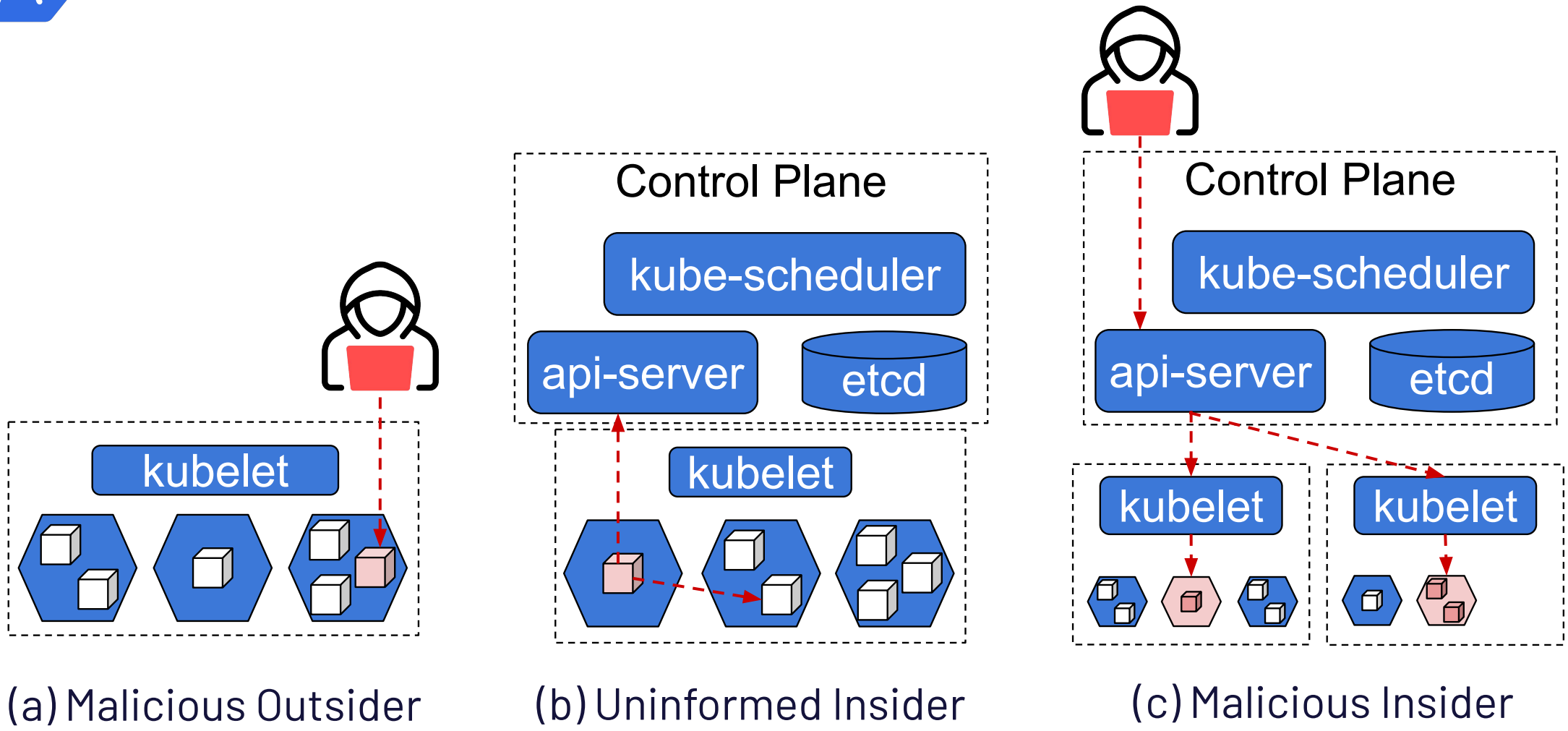
Forensic Readiness

A proactive approach ensuring systems and networks are prepared to efficiently collect, preserve, and analyze evidence when a security incident occurs[1]

Extended Audit Logging ✚ *Automated Checkpointing*

Jason Sachowski, *Implementing Digital Forensic Readiness: From Reactive to Proactive Process*. (2016)

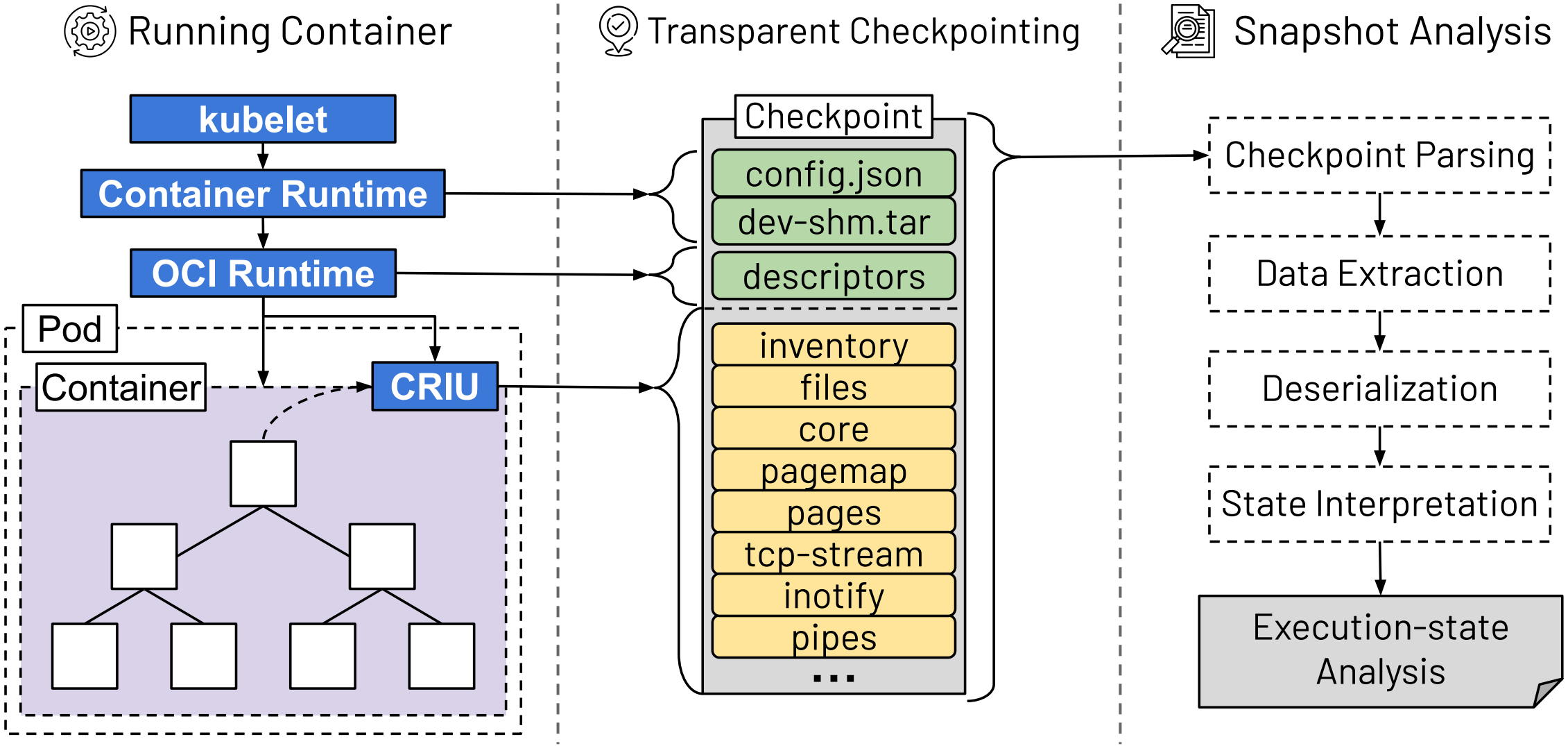
Threat Modeling



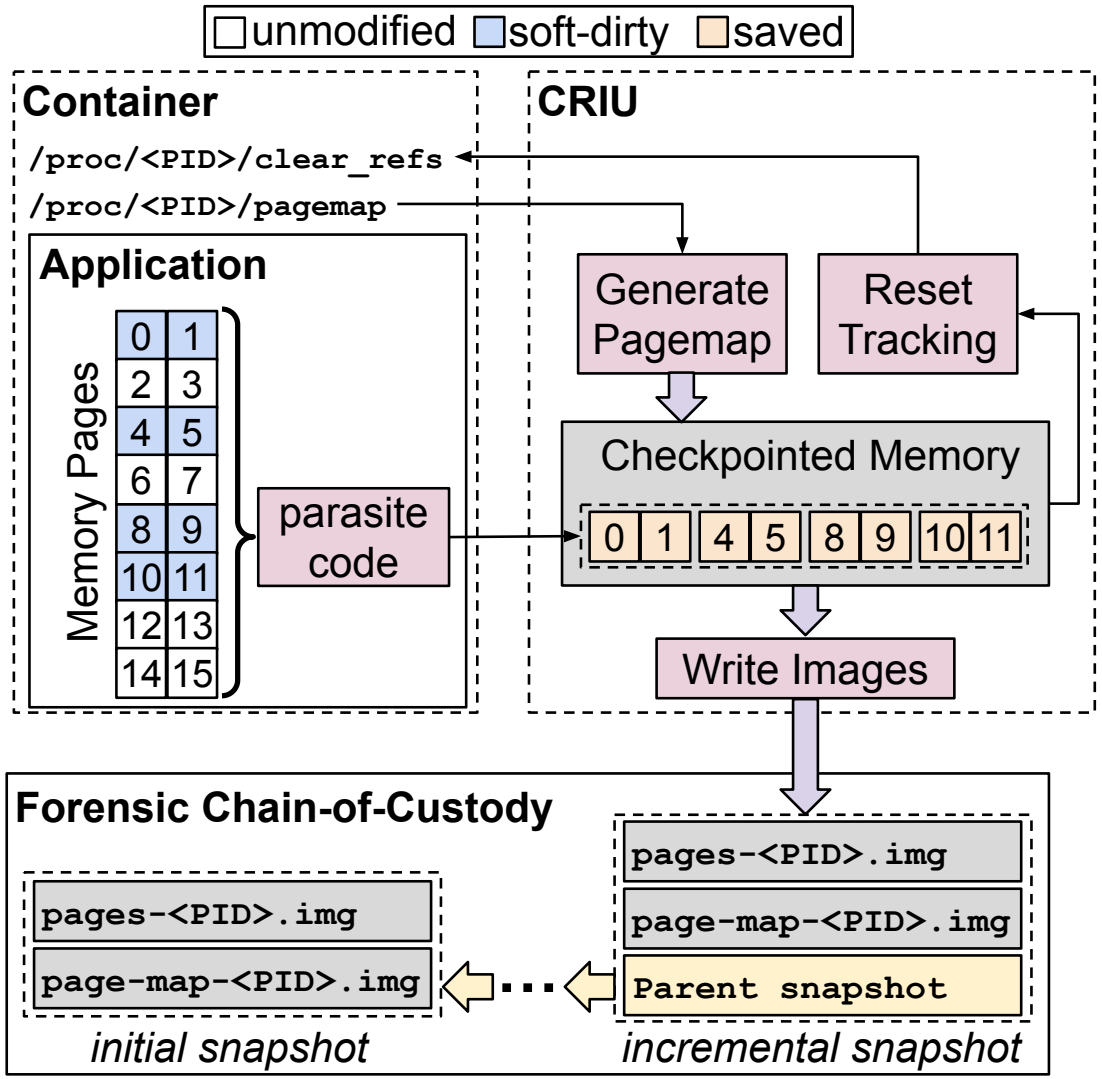
(a) Malicious Outsider (b) Uninformed Insider (c) Malicious Insider

Cloud Native Security Whitepaper (2022)

Snapshot Acquisition



Forensic Snapshot Chain-of-Custody



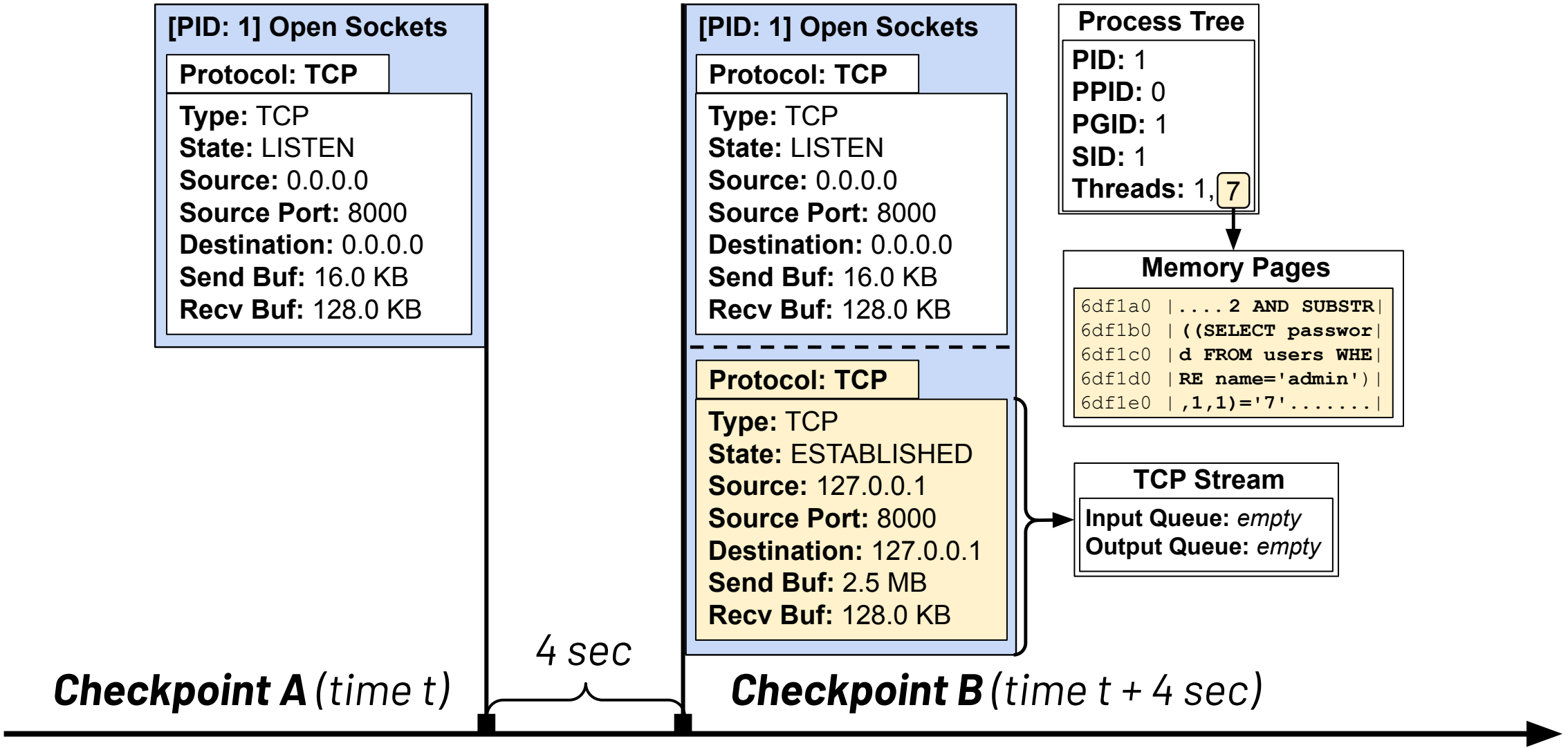
Checkpoints

- Immutable image digests
- Container IDs
- Pod UIDs
- Kubernetes Namespaces

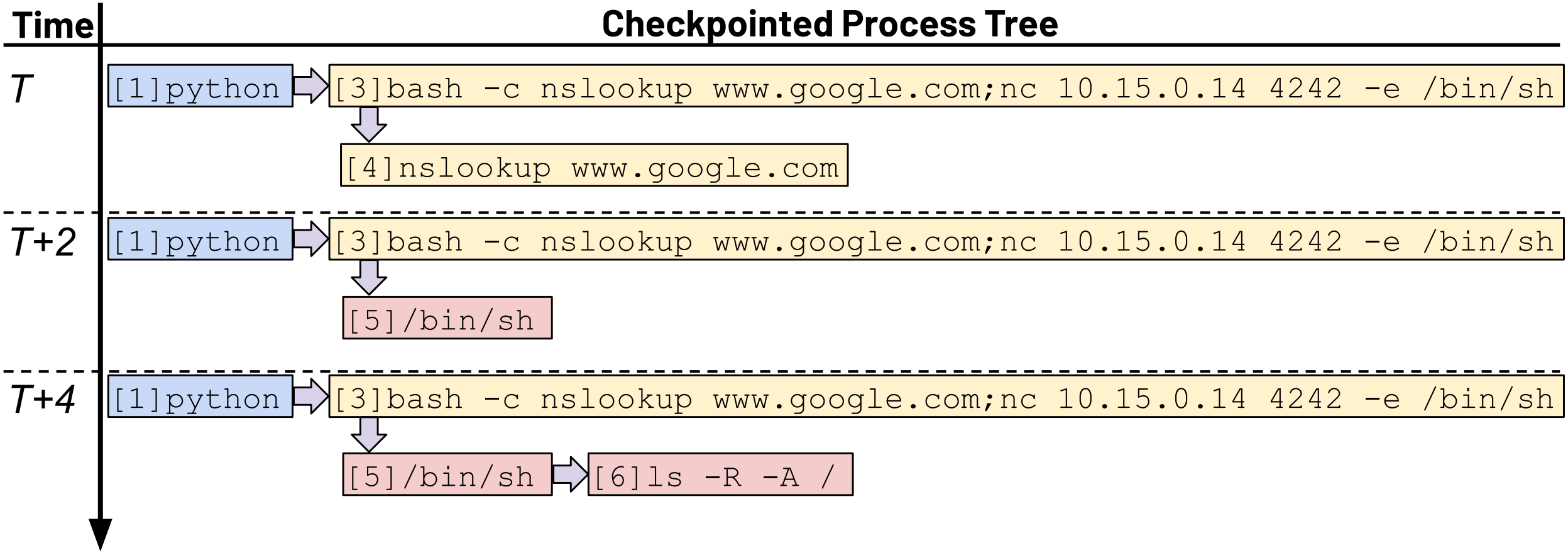
Timestamp Correlation

- Audit logs
- Checkpoint time
- Incident timeline

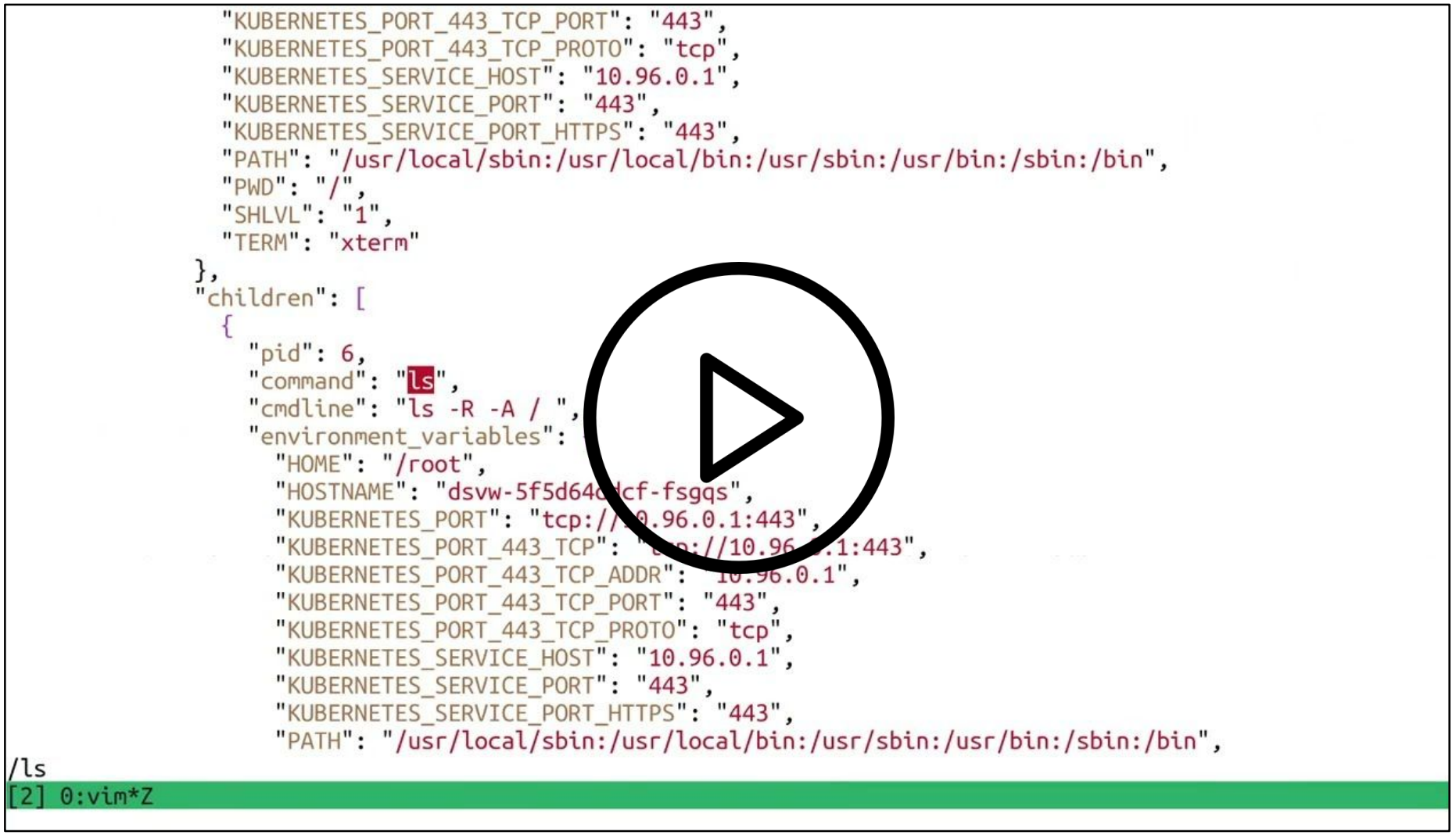
Analysis of SQL Injection Attack



Analysis of Command Injection Attack



Analysis of Command Injection Attack



Key Takeaways

- Detection ≠ Investigation
  - Security alerts detect incidents;
  - Forensic investigations explain *what happened*, *how*, and *why*
- Traditional forensic techniques do not fit Kubernetes
  - Kubernetes = Ephemeral containers + dynamic state-reconciliation
  - Modern malware is memory-resident (fileless)
- Forensic snapshots preserve runtime evidence
  - Snapshots are taken without stopping workloads or alerting attackers
  - Capture memory, processes, and network state from live containers

github.com/checkpoint-restore/checkpointctl

criu.org/Kubernetes

Summary & Questions?

- Kubernetes attacks live in runtime state
- Alerts alone are not evidence
- Snapshots preserve what would otherwise be lost