

Privilege Escalation Detection for Linux Cluster Security

Michael Treaster^{†‡}, Gregory A. Koenig^{†‡},
Xin Meng[†], William Yurcik[†]

[†] National Center for Supercomputing Applications (NCSA)

[‡] Department of Computer Science, University of Illinois at Urbana-Champaign

National Center for Supercomputing Applications



Overview

- Cluster Security
 - Address unique cluster environment
 - Different from security of enterprise resources
 - Monitoring techniques
- Privilege Escalation Detection
 - Detect and identify Unix processes that are running with illegitimately obtained access privileges that were acquired by abusing a vulnerability in a setuid program
 - Address shortcomings of other monitoring techniques

National Center for Supercomputing Applications



Motivation - NVisionCC

- Security tool specifically designed for a cluster environment
- Multifaceted Monitoring
 - Processes
 - File integrity
 - Network ports

Motivation - Cluster Emergent Properties

- *Emergent Properties*
 - Unplanned characteristics or behaviors that arise when simple components are assembled into a more complex system
- Examples:
 - Homogeneity within node classes
 - Login nodes, compute nodes, storage nodes, etc.
 - Coordination with job scheduler
 - Global user list
- Login nodes violate the homogeneity principle
 - Behave more like general-purpose machines
 - Process monitoring is especially ineffective

Privilege Escalation

- Unix security model
 - user ID 0 (root) - Unrestricted access to resources
 - All other users - Limited access to system resources
- `setuid` bit
 - Set as part of the permission bits on an executable file
 - Runs the application with the privileges of the application's owner (usually root), rather than of the user running the application
 - Use of escalated privileges is regulated by the application
- `passwd` example

Unauthorized Privilege Escalation

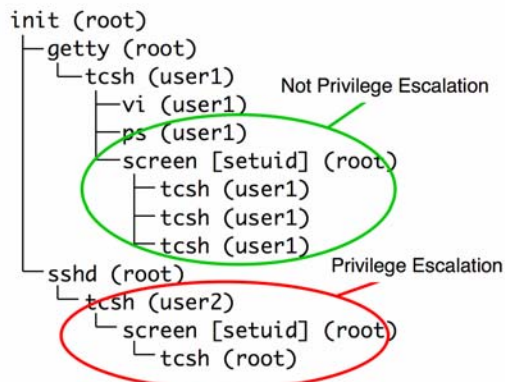
- Provides attacker with unregulated root access to the system
- Attackers often target `setuid` applications
- NVisionCC uses process monitoring to detect the processes spawned by the `setuid` program
 - Relies on process profile

Detection Algorithm

- Check for situations where:
 - Process p has `setuid` bit set
 - $P \rightarrow$ child is owned by user X (usually root)
 - $P \rightarrow$ parent is not owned by user X
- Administrator white list to avoid false positives
- Relies on the global user list emergent property

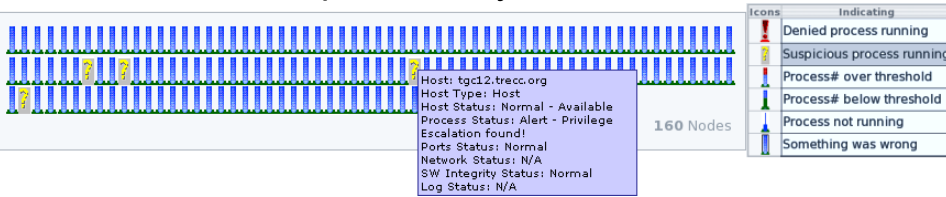
Detection Algorithm (cont)

- Process p is `setuid`
- $P \rightarrow$ child is owned by user X (usually root)
- $P \rightarrow$ parent is not owned by user X

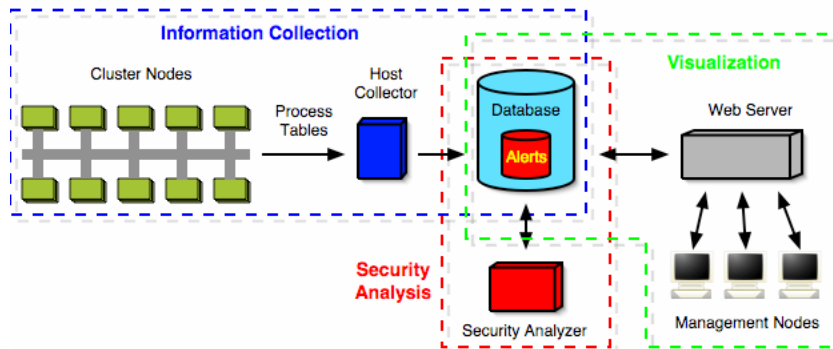


Implementation

- Clumon / NVisionCC plugin
 - Collect process data and store in database
 - Extract data from database
 - Analyze data
 - Insert alerts into database
- Visualization provided by NVisionCC



NVisionCC Architecture



Future Work

- Event-driven, system call monitoring approach
 - Improve monitoring resolution
 - Improve scalability by sending only deltas
- Tree-structured data collection
 - Improve scalability by reducing network load
- Real-world test deployment to assess effectiveness

Conclusions

- NVisionCC is a security tool specifically designed for a cluster environment by leveraging cluster emergent properties.
- NVisionCC fails to adequately protect login nodes, which resemble general purpose machines.
- Privilege escalation detection addresses shortcomings of NVisionCC process monitoring.

Questions?



treaster@ncsa.uiuc.edu

National Center for Supercomputing Applications

