Lecture 23: Reviewing Logs

CS 5430

4/23/2018
Classes of Countermeasures

- **Authentication**: mechanisms that bind principals to actions
- **Authorization**: mechanisms that govern whether actions are permitted
- **Audit**: mechanisms that record and review actions
Uses of audit

- **Individual accountability:** deter misbehavior

- **Event reconstruction:** determine what happened and how to recover

- **Problem monitoring:** real-time intelligence
Audit tasks

• **Recording:**
  - what to log
  - what not to log
  - how to protect the log

• **Reviewing:**
  - manual exploration
  - automated analysis
MANUAL
Manual review

• Enable administrators to explore logs and look for {states, events}

• **Issues:**
  • Designers might not have anticipated the right {states, events} to record
  • Visualization, query, expressivity (HCI/DB issues)
  • Correlation amongst multiple logs
Interfaces

- **Flat text** [example: last time's syslog]
- **Hypertext** [example]
- **DBMS** [example: queries in CMS]
- **Graph** (nodes might be entities like processes and files, edges might be associations like forking or times) [example]
Techniques

- **Temporal replay**: animate what happened when [example]
- **Slice**: display minimal set of log events that affect a given object
Automated review and response

• **Review:** detect suspicious behavior that looks like an attack, or detect violations of explicit policy
  - Custom-built systems
  - Classic AI techniques like training neural nets, expert systems, etc.
  - Modern applications of machine learning

• **Response:** report, take action
INTRUSION DETECTION
Intrusion detection

Intrusion detection system (IDS):
- automated review and response
- responds in (nearly) real time
- components:
  - sensors
  - analysis engine
  - countermeasure deployment
  - audit log
Example: Network monitoring

- **Suspicious behavior:** opening connections to many hosts
- **Automated response:** router reconfigures to isolate suspicious host on its own subnet with access only to (e.g.) virus scanner download, notifies administrators
- **Issue:** errors...
Errors

- **False positive**: raise an alarm for a non-attack
  - makes administrators less confident in warnings
  - perhaps leading to actual attacks being dismissed
- **False negative**: not raise an alarm for an attack
  - the attackers get in undetected!
- Tradeoff between the two needs to be tunable; difficult to achieve the right classification statistics
Identification methodologies

[Denning 1987]

1. **Signature based**: recognize known attacks
2. **Specification based**: recognize bad behavior
3. **Anomaly based**: recognize abnormal behavior
1. Signature-based detection

- A.k.a. *misuse detection* and *rule-based detection*
- Characterize known attacks with signatures
- If behavior ever matches signature, declare an intrusion

**Issues:**
- Works only for known attacks
- Signature needs to be robust w.r.t. small changes in attack
Example: Tripwire

[open source tool and commercial product]

- **Policy**: certain files shouldn't change
- **State snapshot**: analyzes filesystem, stores database of file hashes
- **Automated response**: runs (e.g. daily) and reports change of hash
- **Issues**: where to store database, how to protect its integrity, how to protect tripwire itself?
Example: Snort

```
audit *local.rules x

alert icmp any any -> $HOME_NET any (msg:"ICMP test"; sid:1000001; rev:1; classtype:icmp-event;)

# alert tcp $EXTERNAL_NET any -> $HOME_NET 53 ( msg:"OS-LINUX
OS-LINUX x86 Linux overflow attempt ADMv2";
flow:to_server,established; content:"|89 F7 29 C7 89 F3 89 F9 89 F2 AC|<|FE|",fast_pattern,nocase; metadata:ruleset community;
service:dns; classtype:attempted-admin; sid:265; rev:15; )
```
Network-based IDS

- Typically a separate machine
- **Stealth mode:**
  - one NIC faces the network being monitored, no packets ever sent out on it, no packets can be routed specifically to it
  - another NIC faces a separate network through which alarms are sent
- **Honeypot:**
  - dedicated machines(s) or networks
  - purpose is to look attractive to attacker
  - but actually just a trap: monitored to detect and surveil attacker
2. Specification-based detection

- Characterize good behavior of program with a specification
- If behavior ever departs from specification, declare an intrusion
- **Issues:**
  - Effort to create specifications
  - Any program is a potential vulnerability if executed by a privileged user
Example: Distributed Program Execution Monitor (DPEM)

[Ko et al. 1997]

- Monitors Unix audit logs
- Analyst writes grammar in DSL to describe good behavior
- Parser checks conformance of logs with grammar
- Distributed because it combines information from multiple hosts
3. Anomaly-based detection

- Characterize normal behavior of system
- If behavior ever departs far enough from normal, declare an intrusion

**Issues:**
- Feature identification
- Obtaining data on what is normal
Example: Haystack

[Smaha 1988]

• Monitors value of some statistic of interest over a sliding time window: \( a_i, a_{i+1}, ..., a_j \)
• Determine lower and upper bounds \( t_L \) and \( t_U \) such that 90% of values lie between \( t_L \) and \( t_U \)
• If next value is outside \( t_L \) and \( t_U \), raise an alarm
• Adaptive: as window moves, detector itself adjusts
Statistical models

• Threshold models: min and max
• Moment models: mean and standard deviation
• Markov models: probability of next event based on current state
• Seems like a job for machine learning…
Machine learning

• Despite extensive academic research, “Machine learning [for IDS] is rarely employed in…real world settings” [Sommer & Paxson 2010]
• ML is great for classification: finding similarities
• ML is not as great at outlier detection: here, “normal vs. abnormal”
• ML in adversarial setting not well understood
Identification methodologies

1. **Signature based**: recognize known attacks
2. **Specification based**: recognize bad behavior
3. **Anomaly based**: recognize abnormal behavior
INTRUSION RESPONSE
Intrusion handling

[Northcutt 1998]

1. Preparation
2. Identification
3. Containment
4. Eradication
5. Recovery
6. Follow up
Automated response

- **Monitor**: collect (additional) data
- **Protect**: reduce exposure of system
- **Alert**: call a human
Counterattack

- **Legal:** file criminal complaint
- **Technical:** damage attacker to stop attack or prevent future attacks
  - Might harm an innocent party
  - Might expose you to legal liability