Lecture 14

Auditing and Intrusion Detection

Eliminating the Danger before it Arises

Wholeness Statement

Auditing is a detection mechanism for determining security violations. An intrusion detection system is a form of auditing. Violations of the laws of nature are detected by the individual in the form of unhappiness and stress. TM cultures the individual to act in accord with all the laws of nature and brings about happiness and fulfillment.

Overview

1. Logging records events that can be used to analyze system use
2. Auditing is the process that analyzes logs
3. Intrusion detection is a form of auditing
4. Intrusion response: once an intrusion has been detected, how can the system be protected?
5. Computer security sins: format string problems, integer range errors

Auditing

Chapter 21, pp. 423-450

Definitions

Definition 21-1. *Logging* is the recording of events or statistics to provide information about system use and performance.

Definition 21-2. *Auditing* is the analysis of log records to present information about the system in a clear, understandable manner.

Uses of Auditing

1. Determine sequence of events leading to system being in a non-secure state
2. Determine patterns of usage (are protection mechanisms effective)
3. Deter attacks because attackers know that their activities are logged. (accountability)
4. Record system errors (to help detect non-security problems)

Problems

- What do you log?
  - Hint: looking for violations of a policy, so record at least what will show such violations (e.g., failed login attempts, use of privileges to change security settings)
- What do you audit?
  - Need not audit everything, but should audit things related to the security policy

Anatomy of an Auditing System

- Logger
  - Records information, usually controlled by parameters
- Analyzer
  - Analyzes logged information looking for something
- Notifier
  - Reports results of analysis

Logger

- Type and quantity of information recorded is controlled by system or program configuration parameters (configuration file)
- May be human readable or not
  - If not, usually viewing tools supplied
  - Space available and portability (XML) influence storage format
  - Usually log records go to database, but as a backup can use a file.

Example: Windows NT

- Different logs for different types of events
  - ***System event* logs** record system crashes, component failures, and other system events
  - ***Application event* logs** record events that applications request be recorded
  - ***Security event* log** records security-critical events such as logging in and out, system file accesses, and other events
- Logs are binary; use *event viewer* to see them
- If log is full, can have system shut down, disable logging, or overwrite existing logs

Windows NT Sample Entry

Date: 2/12/2000 Source: Security
Time: 13:03 Category: Detailed Tracking
Type: Success EventID: 592
User: WINDSOR\Administrator
Computer: WINDSOR

Description:
A new process has been created:
New Process ID: 2216594592
Image File Name:
\Program Files\Internet Explorer\IEXPLORE.EXE
Creator Process ID: 2217918496
User Name: Administrator
FDomain: WINDSOR
Logon ID: (0x0,0x14B4c4)

- **[Would be in graphical format]**

Main Point

1. Logging is the recording of events or statistics to provide information about system use and performance. It may seem to be useless overhead to the normal operation of the system but that is far from the case. It's usefulness is brought to light when auditing detects insecure activities from information in the log files. Similarly, it might seem that resting will result in less activity but in reality the most powerful activity is performed by a deeply rested individual. TM provides that deep rest.

Analyzer

- Analyzes one or more logs
  - Logs may come from multiple systems, or a single system
  - May lead to changes in logging
  - May lead to a report of an event
- Logs may need to be sanitized before analysis or may only be analyzed by system administrators.
  - I.E., Logs are something to be protected.

Examples

- Using swatch to find instances of telnet from tcpd logs:

  /telnet/&!/localhost/&!/*.site.com/
- The above regular expression would be in a configuration file.

Examples

- Using swatch to find instances of telnet from tcpd logs:

  /telnet/&!/localhost/&!/*.site.com/
- Matches all log file entries containing the word telnet and not containing either “localhost” or any string ending in “.site.com”

Notifier

- Informs analyst, other entities of results of analysis
- May reconfigure logging and/or analysis on basis of results

Examples

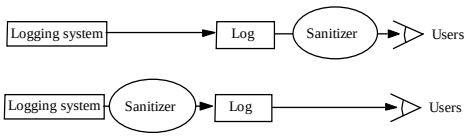
- Using swatch to notify of telnets

  /telnet/&!/localhost/&!/*.site.com/ mail staff
- Three failed logins in a row disable user account
  - Notifier disables account, notifies sysadmin

Log Sanitization

- Let *U* be the set of users,
- Let *P* be the policy defining the set of information *C*(*U*) that *U* cannot see;
- The log is *sanitized* when all information in *C*(*U*) is deleted from log
- Two types of *P*
  - *C*(*U*) can’t leave site
    - People inside site are trusted and information not sensitive to them
  - *C*(*U*) can’t leave system
    - People inside site are not trusted or (more commonly) the information is sensitive to them
    - Don’t log this sensitive information

Logging Organization



- Top prevents information from leaving site
  - Users' privacy not protected from system administrators, other administrative personnel
- Bottom prevents information from leaving system
  - Data simply not recorded, or data scrambled before recording

Audit Browsing

- Goal of browser: present log information in a form easy to understand and use
- Several reasons to do this:
  - Audit mechanisms may miss problems that auditors will spot
  - Mechanisms may be unsophisticated or make invalid assumptions about log format or meaning
  - Logs usually not integrated; often different formats, syntax, *etc*.

Browsing Techniques

- Text display
  - Does not indicate relationships between events
- Hypertext display
  - Indicates local relationships between events
  - Does not indicate global relationships clearly
- Relational database browsing
  - DBMS performs correlations, so auditor need not know in advance what associations are of interest
  - Preprocessing required, and may limit the associations DBMS can make

More Browsing Techniques

- Replay
  - Shows events occurring in order; if multiple logs, intermingles entries
- Graphing
  - Nodes are entities, edges relationships
  - Often too cluttered to show everything, so graphing selects subsets of events
- Slicing
  - Show minimum set of log events affecting object
  - Focuses on local relationships, not global ones

Example: Visual Audit Browser

- Frame Visualizer
  - Generates graphical representation of logs
- Movie Maker
  - Generates sequence of graphs, each event creating a new graph suitably modified
- Hypertext Generator
  - Produces page per user, page per modified file, summary and index pages
- Focused Audit Browser
  - Enter node name, displays node, incident edges, and nodes at end of edges

Summary

- Logging is collecting and recording; audit is analysis
- Need to have clear goals when designing an audit system
- Auditing should be designed into system, not patched into system after it is implemented
- Browsing through logs helps auditors determine completeness of audit (and effectiveness of audit mechanisms!)

Main Point

2. Auditing is the analysis of log records to present information about the system in a clear and understandable manner. In some sense stress might be likened to a log record of some inappropriate past action. However, the analogy breaks done when that stress is released during meditation in the form of thoughts. We do not analyze the thoughts we have during meditation. Just take them as they come.

Intrusion Detection

Chapter 22, pp. 455-484

Principles of Intrusion Detection

- Characteristics of systems NOT under attack
  1. No anomalies: User, process actions conform to statistically predictable pattern
  2. No misuse: User, process actions do not include sequences of actions that subvert the security policy
  3. Specs satisfied: Process actions correspond to a set of specifications describing what the processes are allowed to do
- Systems under attack do not meet at least one of these.
- If an auditing mechanism can detect any of these it is an intrusion detection system.

Example

- Goal: insert a backdoor into a system (a method of bypassing normal authentication)
  - Intruder needs to modify a system configuration file or a program which requires privilege
  - So attacker enters system as an unprivileged user and must acquire privilege
    - Nonprivileged user may not normally acquire privilege (anomaly which violates #1)
    - Attacker may break in using a sequence of commands that violate security policy (misuse which violates #2)
    - Attacker may cause program to act in ways that violate the program's specification (e.g. implanting a virus causing program to violate #3)

Basic Intrusion Detection

- *Attack tool* is automated script designed to violate a security policy
- Example: *rootkit*
  - Includes password sniffer
  - Designed to hide itself using Trojaned versions of various programs (*ps*, *ls*, *find*, *netstat*, etc.)
  - Adds back doors (*login*, *telnetd*, etc.)
  - Has tools to clean up log entries (*zapper, etc.)*

Detection

- *Rootkit* configuration files cause *ls*, *du*, etc. to hide information
  - *ls* lists all files in a directory
    - Except those hidden by configuration file
  - *dirdump* (local program to list directory entries) lists them too
    - Run both and compare counts
    - If they differ, *ls* is doctored
- Other approaches possible

Key Point

- *Rootkit* does *not* alter kernel or file structures to conceal files, processes, and network connections
  - It alters the programs or system calls that *interpret* those structures
  - Find some entry point for interpretation that *rootkit* did not alter
  - The inconsistency is an anomaly (violates #1)

Denning's Model

- Hypothesis: exploiting vulnerabilities requires abnormal use of normal commands or instructions
  - Includes deviation from usual actions
  - Includes execution of actions leading to break-ins
  - Includes actions inconsistent with specifications of privileged programs

Goals of Intrusion Detection System (IDS)

- Detect wide variety of intrusions
  - Previously known and unknown attacks
  - Suggests need to learn/adapt to new attacks or changes in behavior
- Detect intrusions in timely fashion
  - May need to be be real-time, especially when system responds to intrusion
    - Problem: analyzing commands may impact response time of system
  - May suffice to report intrusion occurred a few minutes or hours ago

Goals of IDS (cont.)

- Present analysis in simple, easy-to-understand format
  - Ideally a binary indicator
  - Usually more complex, allowing analyst to examine suspected attack
  - User interface critical, especially when monitoring many systems
- Be accurate
  - Minimize false positives, false negatives
  - Minimize time spent verifying attacks, looking for them

Models of Intrusion Detection

- Anomaly detection
  - What is usual, is known
  - What is unusual, is bad
- Misuse detection
  - What is bad, is known
  - What is not bad, is good
- Specification-based detection
  - What is good, is known
  - What is not good, is bad

Example

- From past experience Annapurna has determined that each student on the average drinks one pint of milk per day.
- Based on this information they budget for X gallons of milk per day.
- If on some day, all X gallons have been used before the midday meal, this is an unusual so there is reason to suspect that someone is stealing milk.
- Which detection model is being used?

Example

- A diner must show a badge or purchase a meal before eating at Annapurna.
- Each diner uses a tray.
- If the number of trays washed in a day does not equal the number of diners counted by the door checker, then somebody has cheated Annapurna and eaten for free.
- Which detection model is being used?

Example

- A student usually accesses only the dining room and the dish room.
- If a student is found in the kitchen this is reported (he/she may be trying to put some chicken in the soup).
- Which detection model is being used?

Anomaly Detection

Anomaly Detection

Definition 22-2. *Anomaly detection* analyzes a set of characteristics of the system, and compares their behavior with expected values; it reports when computed statistics do not match expected statistics

- Threshold metrics
- Statistical moments (skip)
- Markov model (skip)

Threshold Metrics

- Counts number of events that occur
  - Between *m* and *n* events (inclusive) expected to occur
  - If number falls outside this range, anomalous
- Example
  - Windows: lock user out after *k* failed sequential login attempts. Range is (0, *k*–1).
    - *k* or more failed logins deemed anomalous

Difficulties

- Appropriate threshold may depend on non-obvious factors
  - Typing skill of users
  - If keyboards are US keyboards, and most users are French, typing errors very common
    - Dvorak vs. non-Dvorak within the US

Misuse Modeling

Misuse Modeling

- Determines whether a sequence of instructions being executed is known to violate the site security policy
  - Descriptions of known or potential exploits grouped into rule sets
  - IDS matches data against rule sets; on success, potential attack found
- Cannot detect attacks unknown to developers of rule sets
  - No rules to cover them

Note: In some contexts "misuse" refers to an attack by an insider or authorized user. However, in the context of intrusion detection systems, it means "rule-based detection".

Specification Modeling

Specification Modeling

- Determines whether execution of sequence of instructions violates specification
- Only need to check programs that alter protection state of system

Comparison and Contrast

- Anomaly detection: detects unusual events, but these are not necessarily security problems
- Misuse detection: if all policy rules known, easy to construct rule sets to detect violations
  - Usual case is that much of policy is unspecified, so rule sets describe attacks, and are not complete
- Specification-based vs. misuse:
  - spec assumes if specifications are followed, policy not violated;
  - misuse assumes if policy as embodied in rule sets is followed, policy not violated

IDS Architecture

- Basically, a sophisticated audit system
  - *Agent* like logger; it gathers data for analysis
    - Obtains information and sends to director
  - *Director* like analyzer; it analyzes data obtained from the agents according to its internal rules
    - Analyzes information to determine if attack under way
  - *Notifier* obtains results from director, and takes some action
    - May simply notify security officer
    - May reconfigure agents or director to alter collection or analysis methods
    - May activate response mechanism

Intrusion Response

Incident Prevention

- Identify attack *before* it completes
- Prevent it from completing (manually or automatically)
- Jails useful for this
  - Attacker placed in a confined environment that looks like a full, unrestricted environment
  - Attacker may download files, but gets bogus ones
  - Can imitate a slow system, or an unreliable one
  - Useful to figure out what attacker wants
- Honeypots are networks whose only purpose is to detect intruders. If there is any activity on a honeypot, it is an intruder! Ideal way to monitor what an intruder is doing without risking damage to a real system

Intrusion Handling

- Restoring system to satisfy site security policy
- Six phases
  - *Preparation* for attack (beforehand know who to call and what to do when an attack occurs)
  - *Identification* of attack (triggers remaining phases)
  - *Containment* of attack (limit what attacker can access)
  - *Eradication* of attack (stop attack)
  - *Recovery* from attack (restore system to secure state)
  - *Follow-up* to attack (analysis and other actions)

Follow-Up Phase

- Take action external to system against attacker
  - Thumbprinting: traceback at the connection level
  - IP header marking: traceback at the packet level
  - Counterattacking

Counterattacking

- Use legal procedures
  - Collect chain of evidence so legal authorities can establish attack was real
  - Check with lawyers for this
    - Rules of evidence very specific and detailed
    - If you don't follow them, expect case to be dropped
- Technical attack
  - Goal is to damage attacker seriously enough to stop current attack and deter future attacks

Consequences

1. May harm innocent party
   - Attacker may have broken into source of attack or may be impersonating innocent party
2. May have side effects
   - If counterattack is flooding, may block legitimate use of network
3. May make network less usable
   - Counterattack absorbs network resources and makes threats more immediate

Key Points

- Intrusion detection is a form of auditing
- Anomaly detection looks for unexpected events
- Misuse detection looks for what is known to be bad
- Specification-based detection looks for what is known not to be good
- Intrusion response requires careful thought and planning

Security Policy Considerations

- Whenever a site suffers an incident that may compromise computer security, the strategies for reacting may be influenced by two opposing pressures:
  - recover and get back on line as soon as possible (**Protect and Proceed**) or
  - catch the intruder (**Pursue and Prosecute**)
- See section 5 of http://www.faqs.org/rfcs/rfc1244.html

Snort

- Snort written by Marty Roesch to be cross platform (used libcap).
- Released in 1998, first version had 1600 lines of code
- Intially it was just a packet sniffer
- Added signature-based analysis in 1999, now Snort was a lightweight IDS
- Plugins added later (e.g. monitor usage of a particular database)
- Rules are available from snort.org
- See www.snort.org

Irony (Connecting today with Lecture 13)

(1) CRITICAL: Snort DCE-RPC Preprocessor Buffer Overflow

Affected:

Snort versions prior to 2.6.1.3

Snort is shipped as part of many other products including Sourcefire's commercial intrusion prevention/detection products and several Linux distributions.

Irony (cont.)

Description: Snort, a popular open source **intrusion detection** and prevention system, contains a **buffer overflow** in its handling of the DCE-RPC protocol. Microsoft RPC protocol, based on the DCE-RPC reference, is decoded by Snort to detect numerous attacks targeting RPC vulnerabilities. A sequence of specially-crafted DCE-RPC requests could trigger this buffer overflow and execute arbitrary code with the privileges of the Snort process, often root. Since Snort's DCE-RPC preprocessor is enabled by default, attackers can easily send **malicious** traffic on a network segment monitored by Snort to exploit this flaw. The technical details can be obtained via source code analysis.

Main Point

3. An intrusion detection system is a high tech equivalent of a burglar alarm. It is configured to monitor information gateways, hostile activities and known intruders. It is important to have your meditation checked regularly in the first year of meditating to detect the intrusion of effort into the practice.

CONNECTING THE PARTS OF KNOWLEDGE WITH THE WHOLENESS OF KNOWLEDGE

1. Letting users know that logging is being done promotes accountability.
2. The three types of models used by intrusion detection systems are anomaly modeling, misuse modeling, and specification modeling.

3. Transcendental Consciousness is the field of complete balance, safety, and bliss.
4. Wholeness moving within itself: in Unity Consciousness, there can be no enemies because the world is my family; everything is experienced as expressions of my own Self.