Intrusion Detection

- Principles

- Basics

- Models of Intrusion Detection

- Architecture of an IDS

- Organization

- Incident Response

Principles of Intrusion Detection

- Characteristics of systems not under attack
  - User, process actions conform to statistically predictable pattern
  - User, process actions do not include sequences of actions that subvert the security policy
  - 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

Example

- Goal: insert a back door into a system
  - Intruder will modify system configuration file or program
  - Requires privilege; attacker enters system as an unprivileged user and must acquire privilege
    - Nonprivileged user may not normally acquire privilege (violates #1)
    - Attacker may break in using sequence of commands that violate security policy (violates #2)
    - Attacker may cause program to act in ways that violate program's specification

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)

UTD

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 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

- 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

Anomaly Detection

- Analyzes a set of characteristics of system, and compares their values with expected values; report when computed statistics do not match expected statistics
  - Threshold metrics
  - Statistical moments
  - Markov model

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

Statistical Moments

- Analyzer computes standard deviation (first two moments), other measures of correlation (higher moments)
  - If measured values fall outside expected interval for particular moments, anomalous
- Potential problem
  - Profile may evolve over time; solution is to weigh data appropriately or alter rules to take changes into account

UTD

Example: IDES

- **Developed at SRI International to test Denning's model**
  - Represent users, login session, other entities as ordered sequence of statistics $\langle q_{0,j}, \ldots, q_{n,j} \rangle$
  - $q_{i,j}$ (statistic i for day j) is count or time interval
  - Weighting favors recent behavior over past behavior
    - $A_{k,j}$ sum of counts making up metric of kth statistic on jth day
    - $q_{k,l+1} = A_{k,l+1} - A_{k,l} + 2^{-rt}q_{k,l}$ where t is number of log entries/total time since start, r factor determined through experience

Potential Problems

- Assumes behavior of processes and users can be modeled statistically
  - Ideal: matches a known distribution such as Gaussian or normal
  - Otherwise, must use techniques like clustering to determine moments, characteristics that show anomalies, etc.
- Real-time computation a problem too

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

Example: NFR

- Built to make adding new rules easily
- Architecture:
  - Packet sucker: read packets from network
  - Decision engine: uses filters to extract information
  - Backend: write data generated by filters to disk
    - Query backend allows administrators to extract raw, postprocessed data from this file
    - Query backend is separate from NFR process

Comparison and Contrast

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

IDS Architecture

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

UTD

Agents

- Obtains information and sends to director
- May put information into another form
  - Preprocessing of records to extract relevant parts
- May delete unneeded information
- Director may request agent send other information

Example

- IDS uses failed login attempts in its analysis
- Agent scans login log every 5 minutes, sends director for each new login attempt:
  - Time of failed login
  - Account name and entered password
- Director requests all records of login (failed or not) for particular user
  - Suspecting a brute-force cracking attempt

Host-Based Agent

- Obtain information from logs
  - May use many logs as sources
  - May be security-related or not
  - May be virtual logs if agent is part of the kernel
    - Very non-portable
- Agent generates its information
  - Scans information needed by IDS, turns it into equivalent of log record
  - Typically, check policy; may be very complex

Network-Based Agents

- Detects network-oriented attacks
  - Denial of service attack introduced by flooding a network
- Monitor traffic for a large number of hosts
- Examine the contents of the traffic itself
- Agent must have same view of traffic as destination
  - TTL tricks, fragmentation may obscure this
- End-to-end encryption defeats content monitoring
  - Not traffic analysis, though

Network Issues

- Network architecture dictates agent placement
  - Ethernet or broadcast medium: one agent per subnet
  - Point-to-point medium: one agent per connection, or agent at distribution/routing point
- Focus is usually on intruders entering network
  - If few entry points, place network agents behind them
  - Does not help if inside attacks to be monitored

Aggregation of Information

- Agents produce information at multiple layers of abstraction
  - Application-monitoring agents provide one view (usually one line) of an event
  - System-monitoring agents provide a different view (usually many lines) of an event
  - Network-monitoring agents provide yet another view (involving many network packets) of an event

Director

- Reduces information from agents
  - Eliminates unnecessary, redundant records
- Analyzes remaining information to determine if attack under way
  - Analysis engine can use a number of techniques, discussed before, to do this
- Usually run on separate system
  - Does not impact performance of monitored systems
  - Rules, profiles not available to ordinary users

Example

- Jane logs in to perform system maintenance during the day
- She logs in at night to write reports
- One night she begins recompiling the kernel
- Agent #1 reports logins and logouts
- Agent #2 reports commands executed
  - Neither agent spots discrepancy
  - Director correlates log, spots it at once

UTD

Incident Prevention

- Identify attack *before* it completes
- Prevent it from completing
- 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
  - MLS systems provide natural jails

UTD

Intrusion Handling

- Restoring system to satisfy site security policy
- Six phases
  - *Preparation* for attack (before attack detected)
  - *Identification* of attack
    - *Containment* of attack (confinement)
    - *Eradication* of attack (stop attack)
  - *Recovery* from attack (restore system to secure state)
    - *Follow-up* to attack (analysis and other actions)
  - Discussed in what follows

Containment Phase

- Goal: limit access of attacker to system resources
- Two methods
  - Passive monitoring
  - Constraining access

Passive Monitoring

- Records attacker's actions; does *not* interfere with attack
  - Idea is to find out what the attacker is after and/or methods the attacker is using
- Problem: attacked system is vulnerable throughout
  - Attacker can also attack other systems
- Example: type of operating system can be derived from settings of TCP and IP packets of incoming connections
  - Analyst draws conclusions about source of attack

Constraining Actions

- Reduce protection domain of attacker
- Problem: if defenders do not know what attacker is after, reduced protection domain may contain what the attacker is after
  - Stoll created document that attacker downloaded
  - Download took several hours, during which the phone call was traced to Germany

Deception

- ## Deception Tool Kit
  - Creates false network interface
  - Can present any network configuration to attackers
  - When probed, can return wide range of vulnerabilities
  - Attacker wastes time attacking non-existent systems while analyst collects and analyzes attacks to determine goals and abilities of attacker
  - Experiments show deception is effective response to keep attackers from targeting real systems

Eradication Phase

- Usual approach: deny or remove access to system, or terminate processes involved in attack
- Use wrappers to implement access control
  - Example: wrap system calls
    - On invocation, wrapper takes control of process
    - Wrapper can log call, deny access, do intrusion detection
    - Experiments focusing on intrusion detection used multiple wrappers to terminate suspicious processes
  - Example: network connections
    - Wrapper around servers log, do access control on, incoming connections and control access to Web-based databases

UTD

Firewalls

- Mediate access to organization's network
  - Also mediate access out to the Internet
- Example: Java applets filtered at firewall
  - Use proxy server to rewrite them
    - Change "<applet>" to something else
  - Discard incoming web files with hex sequence CA FE BA BE
    - All Java class files begin with this
  - Block all files with name ending in ".class" or ".zip"
    - Lots of false positives

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

UTD

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. Antithetical to shared use of network
   - Counterattack absorbs network resources and makes threats more immediate

4. May be legally actionable

Example: Counterworm

- Counterworm given signature of real worm
  - Counterworm spreads rapidly, deleting all occurrences of original worm
- Some issues
  - How can counterworm be set up to delete *only* targeted worm?
  - What if infected system is gathering worms for research?
  - How do originators of counterworm know it will not cause problems for any system?
    - And are they legally liable if it does?

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