Intrusion Detection

- Principles
- Basics
- Models of Intrusion Detection
- Architecture of an IDS
- Organization

1

Definitions

- Intrusion
  - A set of actions aimed at compromising the security goals of a computing and networking resource
    - Integrity, confidentiality, availability
- Intrusion detection
  - The process of identifying and responding to intrusion activities

2

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 (or are not) allowed to do
- Systems under attack do not meet at least one of these characteristics

3

D.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

4

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 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

5

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

6

Assumptions

- Primary assumptions:
  - System activities are observable
  - Normal and intrusive activities have distinct evidence
- Components of intrusion detection systems:
  - From an algorithmic/model perspective:
    - Features - capture intrusion evidences
    - Analysis - piece evidences together
  - From a system architecture perspective:
    - Audit data processor, knowledge base, detection engine, decision engine, action (alarm generation and responses)

7

Approaches

- Modeling
  - Features: evidence extracted from audit data
  - Analysis: piecing the evidences together
    - Misuse detection (rule-based approach)
    - Anomaly detection (statistical-based approach)
- Deployment
  - Network-based
  - Host-based
- Development and maintenance
  - Hand-coding of "expert" knowledge
  - Learning based on audit data

8

Models of Intrusion Detection

1. Anomaly detection
   - What is usual, is known
   - What is unusual, is bad
2. Misuse detection
   - What is bad, is known
   - What is not bad, is good
3. Specification-based detection
   - What is good, is known
   - What is not good, is bad

9

1. 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

10

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 NT 4.0: 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

11

Statistical Moments

- Analyzer computes mean and 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

12

Example: IDES

- Developed at SRI International to test Denning's model
  - Represent subjects (users, login session, others) as ordered sequence of statistics $<q_{0,j}, \ldots, q_{n,j}>$
  - $q_{i,j}$ (statistic i for day j) is count or time interval; profile updated daily
  - 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

13

Potential Problems

- Assumes behavior of processes and users can be modeled statistically
  - IDES assumes Gaussian distribution of events
    - Experience indicates not right distribution
  - Otherwise, must use techniques like clustering to determine moments, characteristics that show anomalies, etc. Clustering
    - Does not assume *a priori* distribution of data
    - Obtain data, group into subsets (*clusters*) based on some property (*feature*)
    - Analyze the clusters, not individual data points
- Real-time computation a problem too

14

Markov Model

- Past state affects current transition
- Anomalies based upon *sequences* of events, and not on occurrence of single event
  - Over time, probability of transition developed
  - When transition with low probability occurs, event causing it considered anomalous
- Problem: need to train system to establish valid sequences
  - Use known training data that is not anomalous
  - The more training data, the better the model
  - Training data should cover *all* possible normal uses

15

Example: TIM

- Time-based Inductive Learning (Teng 1990)
- Learning
  - Training data is *abcdedeabcabc*
  - TIM derives following rules:

    R_1: $ab \rightarrow c$ (1.0) R_2: $c \rightarrow d$ (0.5) R_3: $c \rightarrow a$ (0.5)
    R_4: $d \rightarrow e$ (1.0) R_5: $e \rightarrow a$ (0.5) R_6: $e \rightarrow d$ (0.5)
- Detecting
  - Seen: *abd* triggers alert
    - *c* always follows *ab* in rule set
  - Seen: *acf* no alert as multiple events can follow *c*
    - May add rule R_7: $c \rightarrow f$ (0.33) and adjust R_2, R_3

16

Potential Problems of Anomaly Detection

- False Positive: Anomaly activities that are not intrusive are classified as intrusive.
- False Negative: Intrusive activities that are not anomalous result in false negatives, that is events are not flagged intrusive, though they actually are.
- Computational expensive because of the overhead of keeping track of, and possibly updating several system profile metrics.

2. 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: IDIOT

- Event is a single action, or a series of actions, resulting in a single record and change of state
- Five categories of attacks:
  - *Existence*: attack creates file or other entity
  - *Sequence*: attack causes several events sequentially
  - *Partial order*: attack causes 2 or more sequences of events, and events form partial order under temporal relation
  - *Duration*: something exists for interval of time
  - *Interval*: events occur exactly *n* units of time apart

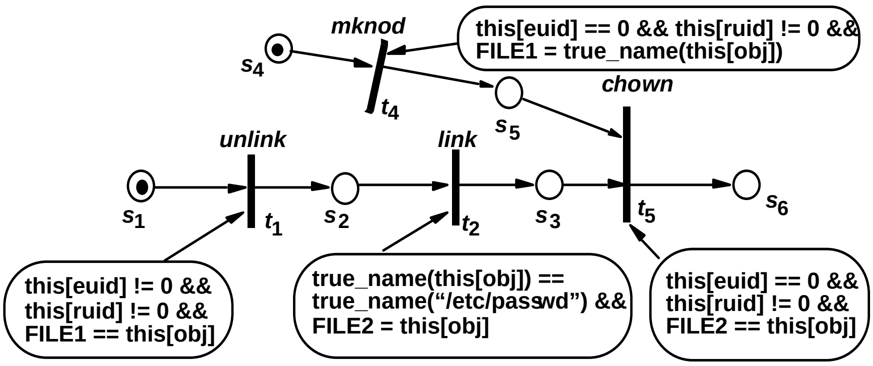
19

IDIOT Representation

- Sequences of (attack) events may be interlaced with other events
- Use colored Petri nets to capture this
  - Each signature corresponds to a particular Colored Petri Automaton
  - Nodes are tokens; edges are transitions
  - Final state of the signature is compromised state
- Example: *mkdir* attack
  - Edges protected by guards (expressions)
  - Tokens move from node to node as guards satisfied

20

IDIOT Analysis



21

IDIOT Features

- New signatures can be added dynamically
  - Partially matched signatures need not be cleared and re-matched (info kept in state)
- Ordering the CPAs allows you to order the checking for attack signatures
  - Useful when you want a priority ordering
  - Can order initial branches of CPA to find sequences known to occur often

22

3. Specification Modeling

- Determines whether execution of sequence of instructions violates specification
- Only need to check programs that alter the protection state of system (potentially critical code).
  - ANY program executed by a privileged user is a potential security threat
- A formalization of what *should* happen (detects unknown attacks)
- Extra effort in analyzing program and specifying its behavior

23

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 violations
- Specification-based vs. misuse: spec assumes if specifications followed, policy not violated; misuse assumes if policy as embodied in rulesets followed, policy not violated
    - Spec-based=per-program, local
    - Misuse=site policy

24

Key Performance Metrics

- Algorithm/Model
  - Alarm: A ; Intrusion: I
  - Detection (true alarm) rate: P(A|I)
    - False negative rate P(¬A|I)
  - False alarm rate: P(A|¬I)
    - True negative rate P(¬A|¬I)
- Architecture
  - Scalable
  - Resilient to attacks

IDS Problem: *Base Rate Fallacy*

- IDS useless unless accurate
  - Significant fraction of intrusions detected
  - Significant number of alarms correspond to intrusions
- Assume 99% accuracy of intrusions detection system
  - 1% of non-intrusions generate alarm
  - 100 in 10,000 events are really intrusions
- Alarm sounds: is it a “real” intrusion?

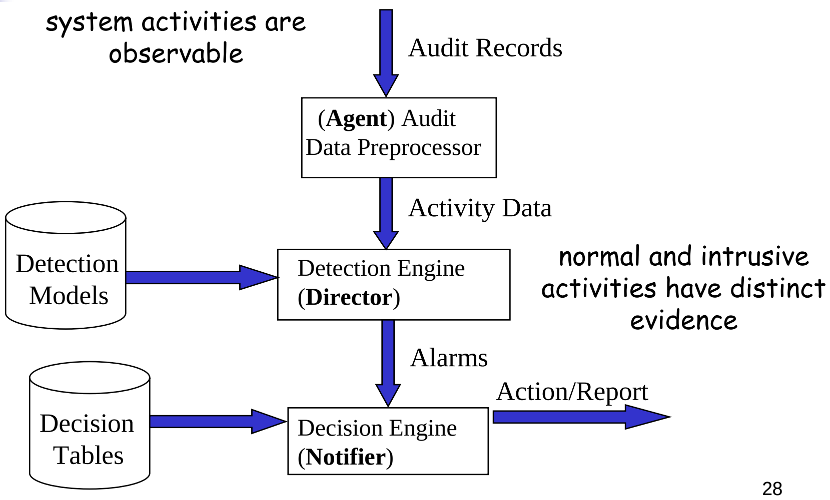
What if only 1 in 10,000 events is an intrusion?

IDS Architecture

- Basically, a sophisticated audit system
  - *Agent* gathers data for analysis
  - *Director* 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

27

Components of an IDS



28

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 to send other information

29

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

30

Host-Based Agent

- Obtain information from logs
  - May use many logs as sources
  - May be security-related or not (accounting)
  - May be virtual logs if agent is part of the kernel
    - Very non-portable
- Agent may generate its information
  - Scans information needed by IDS, turns it into equivalent of log record
  - May generate own info. From state of system, typically for checking policy; may be very complex

Network Intrusion Detection

- Some types of attacks cannot be detected by examining only host-based data, for instance:
  - Doorknob rattling (e.g., password guessing)
  - Masquerading/Spoofing
  - Diversionary attacks (e.g., blatant and subtle attacks)
  - Multipronged attacks (e.g., from multiple sources)
  - Chaining (to make tracing difficult)
  - Loopback (including change of UID)

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
- End-to-end encryption defeats content monitoring
  - Not traffic analysis, though

33

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

34

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

35

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

36

Adaptive Directors

- Modify profiles, change rule sets to adapt their analysis to changes in system
  - Usually use machine learning or planning to determine how to do this
- Example: use neural nets to analyze logs
  - Network adapted to users' behavior over time
  - Used learning techniques to improve classification of events as anomalous
    - Reduced number of false alarms

37

Notifier

- Accepts information from director
- Takes appropriate action
  - Notify system security officer
  - Respond to attack
- Often GUIs
  - Well-designed ones use visualization to convey information

38

Types of Intrusion Detection Systems

- Network-Based Intrusion Detection Systems
  - Have the whole network as the monitoring scope, and monitor the traffic on the network to detect intrusions.
  - Can be run as an independent standalone machine where it promiscuously watches over all network traffic,
  - Or just monitor itself as the target machine to watch over its own traffic. (SYN-flood or a TCP port scan)

39

Types of Intrusion Detection Systems

- Host-based Intrusion Detection Systems (HIDS)
  - Misuse is not confined only to the "bad" outsiders but within organizations.
  - Local inspection of systems is called HIDS to detect malicious activities on a single computer.
  - Monitor operating system specific logs including system, event, and security logs on Windows systems and syslog in Unix environments to monitor sudden changes in these logs.
  - They can be put on a remote host.

40

Advantage of NIDS

- Ability to detect attacks that a host-based system would miss because NIDSs monitor network traffic at a transport layer.
- Difficulty to remove evidence compared with HIDSs.
- Real-time detection and response. Real time notification allows for a quick and appropriate response.
- Ability to detect unsuccessful attacks and malicious intent.

41

Disadvantages of NIDS

- Blind spots. Deployed at the border of an organization network, NIDS are blink to the whole inside network.
- Encrypted data. NIDSs have no capabilities to decrypt encrypted data.

42

Advantages of HIDS

- Ability to verify success or failure of an attack quickly because they log continuing events that have actually occurred, have less false positive than their cousins.
- Low level monitoring. Can see low-level activities such as file accesses, changes to file permissions, attempts to install new executables or attempts to access privileged services, etc.
- Almost real-time detection and response.
- Ability to deal with encrypted and switched environment.
- Cost effectiveness. No additional hardware is needed to install HIDS.

43

Disadvantages of HIDS

- Myopic viewpoint. Since they are deployed at a host, they have a very limited view of the network.
- Since they are close to users, they are more susceptible to illegal tempering.

44

Combining Sources: DIDS

- Neither network-based nor host-based monitoring sufficient to detect some attacks
  - Attacker tries to telnet into system several times using different account names: network-based IDS detects this, but not host-based monitor
  - Attacker tries to log into system using an account without password: host-based IDS detects this, but not network-based monitor
- DIDS uses agents on hosts being monitored, and a network monitor
  - DIDS director uses expert system to analyze data

45

Attackers Moving in Network

- Intruder breaks into system A as *alice*
- Intruder goes from A to system B, and breaks into B's account *bob*
- Host-based mechanisms cannot correlate these
- DIDS director could see *bob* logged in over *alice*'s connection; expert system infers they are the same user
  - Assigns *network identification number* NID to this user

46

Handling Distributed Data

- Agent analyzes logs to extract entries of interest
  - Agent uses signatures to look for attacks
    - Summaries sent to director
  - Other events forwarded directly to director
- DIDS model has agents report:
  - Events (information in log entries)
  - Action, domain

Actions and Domains

- Subjects perform actions
  - session_start, session_end, read, write, execute, terminate, create, delete, move, change_rights, change_user_id
- Domains characterize objects
  - tagged, authentication, audit, network, system, sys_info, user_info, utility, owned, not_owned
  - Objects put into highest domain to which it belongs
    - Tagged, authenticated file is in domain tagged
    - Un-owned network object is in domain network

More on Agent Actions

- Entities can be subjects in one view, objects in another
  - Process: subject when changes protection mode of object, object when process is terminated
- Table determines which events sent to DIDS director
  - Based on actions, domains associated with event
  - All NIDS events sent over so director can track view of system
    - Action is *session_start* or *execute*; domain is *network*

Intrusion Response

If an intrusion is detected, how to protect the system.

- Goal:
  - Minimize the damage of attack
  - Thwart intrusion
  - Attempt to repair damages
- Phases
  - Incident Prevention
  - Intrusion Handling
    - Containment Phase
    - Eradication Phase
    - Follow-Up phase

Incident Prevention

- Identify attack *before* it completes, ideally
- 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
  - Multilevel secure systems are excellent places to implement jails.

51

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

52

Containment Phase

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

53

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

54

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

55

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

56

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

57

IDS Tools

- Snort
- Honeypot, www.honeyd.org
  - A honeypot is a system designed to look like something that an intruder can hack.
  - The goal is to deceive intruders and learn from them without compromising the security of the network.
- IPAudit,

58

Categories of IDSs

There are several ways to distinguish/classify IDS:

- Is the system *dynamic* or *static* ?
  - i.e., does it continuously gather data, or look for snapshots
- Is the system misuse- or specification- or anomaly-based?
  - knows what ‘unacceptable’ looks like, or what ‘acceptable’ looks like?
- Is the system integrated with defenses, primarily investigatory, or used for retaliation?
- Is the system based on *rules* (describe what is intrusive), or on *statistics* (measure deviations from standard)?
- Is the data gathered from the host, the network, or a combination?

59

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

60