Intrusion Detection (IDS)

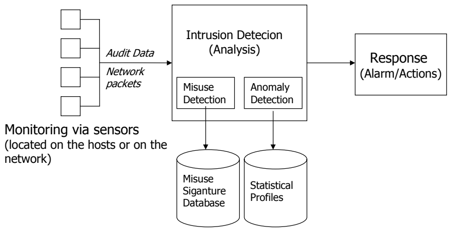
Simone Fischer-Hübner

Motivation for IDS

- Developing absolutely secure systems is not possible
  - Most existing systems have security flaws
  - Abuses by privileged insiders are possible
  - Not all kinds of intrusions are known
- Quick detection of intrusions can help to identify intruders and limit damage
- IDS serves as a deterrent

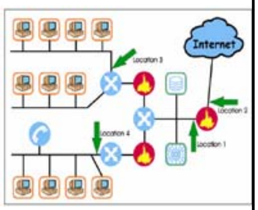
Intrusion Detection (IDS) – Basic concepts

IDS: Software and/or hardware systems monitoring a system, analysing it for signs of security intrusions and eventually triggering response



Sensors - Network based IDS

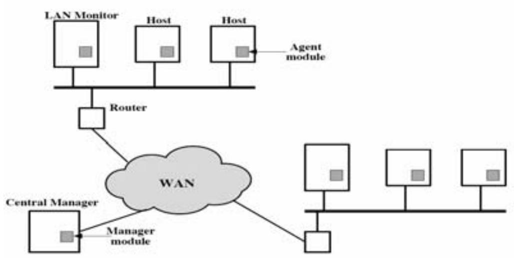
- Capturing and analysing network packets
- Placed at various points in a network
  - Behind the external firewall (Location 1)
  - Outside the external firewall (Location 2)
  - On major network backbones (Location 3)
  - On critical subnets (Location 4)



Sensors – Host based IDS

- Information is collected within an individual computer or application
- Installed on critical hosts
- Audit data are collected on OS level (system logs) and/or application level

Distributed Intrusion Detecion – Architecture Example (centralised)



Distributed Intrusion Detection – Issues

- A distributed intrusion detection system may need to deal with different audit record formats
- One or more nodes in the network will serve as collection and analysis points for the data, which must be securely transmitted to them
- Architecture can be:
  - centralized (single point of analysis, easier but bottleneck) or
  - decentralized (multiple centers that must be coordinated)

Anomaly Detection

- Based on the hypothesis that intrusions can be detected by monitoring a system for abnormal patterns of system usage
- **IDES** (Intrusion Detection Expert System) developed by D.Denning at SRI/International in 1986
- Usually rule-based pattern matching system which includes
  - Statistical profiles for representing the behavior of subjects with respect to objects
  - Rules matching new audit records against profiles, acquire/update profiles, detect anonalous behavior

Examples for anomalies for intrusions:

- Attempted break-ins:
  - abnormally high rate of password failure
- Masquerading, successful break-ins
  - different login time, location or connection type,
  - different accesses to data, execution of programs
- Penetration by legitimate users:
  - login at unusual times,
  - route data to remote printers not normally used,
  - execution of different programs, more protection violations,
  - access to commands/files not normally permitted to him/her
- Viruses:
  - infected program needs more memory, disk space, CPU-time, I/O-activities,
  - it modifies other executable code not normally done by it,
  - increase in the frequency of executable files rewritten in the infected system

IDES Anomaly Detection – Audit Records

Generated by the target system, translated into standard format, transmitted to the IDES system for analysis

Audit record structure:
(subject, action, object, exception-condition, resource-usage, time-stamp)

Decomposition of activities involving multiple objects to single-object actions:
e.g.: COPY GAME.EXE to <LIBRARY>GAME.EXE issued by Smith is aborted, because he does not have write-permission to <LIBRARY>

Audit Records:
(Smith, execute, <Library>COPY.EXE, 0, CPU=0002, 1105821678)
(Smith, read, <Smith>GAME.EXE, 0, RECORDS=0, 1105821679)
(Smith, write, <Library>GAME.EXE, write-viol, Records=0, 1105821679)

IDES Anomaly Detection – Statistical Profiles (I)

- Profiles characterize the behaviour of a subject with respect to an object in terms of a ***statistical metric*** and ***model***
- **Metric:**
  - Random variable x representing a quantitative measure accumulated over a period (period: fixed or time between 2 events)

Examples of types of metrics:

- ***Event counter:***
  - x is the number of audit records satisfying some property occurring during a period, e.g. number of logins during one hour, number of execution failures during one session
- ***Interval timer:***
  - x is the length of time between two related events, e.g. time length between successive logins into one account
- ***Resource measure:***
  - x is the quantity of resources consumed by some action during a period, e.g. number of pages printed per day

IDES Anomaly Detecion – Statistical Profiles (II)

- **Statistical Model:**
  - Given a metric for a random variable x and n observations $x_1,\ldots,x_n$.
  - The statistical model shall determine whether a new observation x_{n+1} is abnormal with respect to the previous observations.
- ***Operational Model:***
  - Abnormality is detected by comparing a new observation of x against fixed limits, e.g. limitation of number of password failures during a short period.
- ***Mean and Standard Deviation Model:***
  - A new observation of x is defined to be abnormal, if it falls outside a confidence interval:

    mean $\pm$ d * stdev *(the probability of a value falling outside this interval is at most 1/d2).*

    sum = $x_1 + x_2 + \ldots + x_n$
    sumsquares = $x_1^2 + \ldots + x_n^2$
    mean = sum / n
    stdev = $\sqrt{\text{(sumsquares / (n-1)} - \text{mean}^2)}$

IDES Example Profile Structure

Profile structure:

(name,
subject-pattern,
action-pattern,
object-pattern,
exception-pattern,
resource-usage-pattern,
period,
metric,
statistical-model,
value,
threshold)

Example of patterns:

Subject patterns: ´Smith´, * → user
Object patterns: ´<Library>*´ , IN(GAME.EXE,EDITOR.EXE)

Anomaly Detection – Examples of Metric/Model Combinations in Profiles

Login Frequency (event counter, mean/ standard deviation model)

Location Frequency (event counter, mean/ standard deviation model)

Session Output (resource measure, mean/ standard deviation model)

Password Fails (event counter, operational model)

Execution Frequency (event counter, mean/standard deviation model)

Execution Denied (event counter, operational model)

Read-, Write,- Delete-Frequency (event counter, mean/standard deviation model)

Read-, Write, Delete-Fails (event counter, operational model)

File Resource Exhaustion (event counter, operational model)

IDES Anomaly Detection – Pattern Matching Rules

Rule-Structure: Condition --> Action-Body

Audit Record Rules:

Condition: A new audit record matches a profile

Body: update of the profile, checking for anomalous behaviour, (generation of an anomaly record, if an abnormality is detected)

Periodic Activity Update Rules:

Condition: The system clock implies a period of length p completes, the period component of a profile is p

Body: update of the matching profile, checking for anomalous behavior, (generation of an anomaly record, if a abnormality is detected)

Anomaly Detection – Pros & Cons

+

- Can detected an attack without previous knowledge about it
- Can deliver the base for signature generation

-

- Requires sophisticated mathematical analysis which is time intensive
- Produces a large number of false alarms
- Requires extensive training sets for the system
- Vulnerable to attacks based on slow change of behavior
- Affects privacy of users

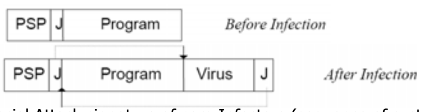
Analysis – Misuse detection

- System activities are scanned for attack signatures, i.e. patterns of network traffic or activities in log files indicating malicious behavior
- Examples:
  - patterns of bits in an IP packet indicating a buffer overflow
  - certain types of TCP SYN packets indicating a SYN flooding attack
  - Sequence of action typical for computer viruses
- Majority of commercial-based IDS products are based on misuse detection
- SNORT is a popular open-source Network Misuse detection based IDS tool (www.snort.org)

Misuse Detection- Example: Buffer overflow attack signatures

- An exec system call audit records for a buffer overflow has the following pattern:
  - The exec call concerns a setuid program, i.e. the *effective user id* and the *real user id* fields are different
  - The argument passed to the exec call is relatively long, making the length of the entrire audit record significantly exceed the length al almost all normal *setuid exec* call
    - Buffer overflow atatcks typcially produce exec audit records with a length > 500 bytes. Only 0.15 % of normal exec audit records are longer than 400 bytes.
  - The exec argument contain opcode in the range of ascii control characters

Misuse Detection – Example: Virus signature



- Typcial Attack signature of com-Infectors (sequence of system calls):
  - Open executable (.com) file to be infected
  - Get date of last modification
  - Get time of last modification
  - Read first 3 bytes to get jump address
  - Go to end of file
  - Append code (of the virus)
  - Go to beginning of the file
  - Write new jump address (3 bytes)
  - Reset date and time
  - Close file

Advantage/Disadvantage

+

- The ratio between detection to false alarm is acceptable
- By counting the occurrence of patterns protective measurements can be applied
- Is not (so much) dependent on the qualification of the maintainers

-

- Can only detect previously known attacks which requires huge databases of attack patterns
- Small variations in an attack can make the attack undetectable

Honeypots

- decoy systems to lure attackers
  - away from accessing critical systems
  - to collect information of their activities
  - to encourage attacker to stay on system so administrator can respond
- are filled with fabricated information
- instrumented to collect detailed information on attackers activities
- single or multiple networked systems
- cf IETF Intrusion Detection WG standards

Anomaly Detection – Examples for Statistical Models (I)

Measure	Model	Type of Intrusion Detected
	Login and Session Activity	
Login frequency by day and time	Mean and standard deviation	Intruders may be likely to log in during off-hours.
Frequency of login at different locations	Mean and standard deviation	Intruders may log in from a location that a particular user rarely or never uses.
Time since last login	Operational	Break-in on a "dead" account.
Elapsed time per session	Mean and standard deviation	Significant deviations might indicate masquerader.
Quantity of output to location	Mean and standard deviation	Excessive amounts of data transmitted to remote locations could signify leakage of sensitive data.
Session resource utilization	Mean and standard deviation	Unusual processor or I/O levels could signal an intruder.
Password failures at login	Operational	Attempted break-in by password guessing.
Failures to login from specified terminals	Operational	Attempted break-in.

Anomaly Detection – Examples for Statistical Models (II)

	Command or Program Execution Activity	
Execution frequency	Mean and standard deviation	May detect intruders, who are likely to use different commands, or a successful penetration by a legitimate user, who has gained access to privileged commands.
Program resource utilization	Mean and standard deviation	An abnormal value might suggest injection of a virus or Trojan horse, which performs side-effects that increase I/O or processor utilization.
Execution denials	Operational model	May detect penetration attempt by individual user who seeks higher privileges.

Anomaly Detection – Examples for Statistical Models (II)

	File access activity	
Read, write, create, delete frequency	Mean and standard deviation	Abnormalities for read and write access for individual users may signify masquerading or browsing.
Records read, written	Mean and standard deviation	Abnormality could signify an attempt to obtain sensitive data by inference and aggregation.
Failure count for read, write, create, delete	Operational	May detect users who persistently attempt to access unauthorized files.