Computer Intrusion Detection

Lecture 6

Basics of Analysis Schemes

Xiangyang Li

Outline

Partially based on R. Bace's book and M. Bishop's book (Computer Security: Art and Science) with some materials courtesy of M. Bishop

- Basic Concepts of Analysis and Detection
- What are attacks?
- Misuse Detection vs. Anomaly Detection
- Hybrid Architecture and Others
- Additional Issues

Goals of Detection

- 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
  - Problem: analyzing commands may impact response time of system
  - May suffice to report intrusion occurred a few minutes or hours ago

Goals of Detection (cont.)

- Present analysis in simple, easy-to-understand format
  - 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

Full Range of Intrusion Analysis

The human detector

- Essential although limited value

External events

- Triggers, e.g. hiring/ firing, anomaly reports, penetration test results, discovery of missing info.

Precursors to intrusion

- Signs of intrusion, e.g. trojans, unauthorized accounts.

Artifacts of intrusion

- Evidence of past intrusions, e.g. system failures, damaged files, or incidental outcomes like abnormal resource use.

Additional Relevant Terms

Feature Selection refers to the process to find or construct most discriminating (informative) attributes.

- New attributes used in analysis

Pattern Recognition refers to the task to extract patterns (using features) that define various types of events.

Data Mining refers to the process of identifying and utilizing such patterns to aggregate events into different groups (classes) or find the corresponding group of an event.

- Clustering
- Classification
- Link analysis
- Sequence analysis

Machine Learning emphasizes automated algorithms to support the above tasks.

Outline

- Basic Concepts of Analysis and Detection
- What are attacks?
- Misuse Detection vs. Anomaly Detection
- Hybrid Architecture and Others
- Additional Issues

Characteristics of Systems NOT under Attack

1. User, process actions conform to statistically predictable pattern.
2. User, process actions do not include sequences of actions that subvert the security policy.
3. Process actions correspond to a set of specifications describing what the processes are allowed to do.

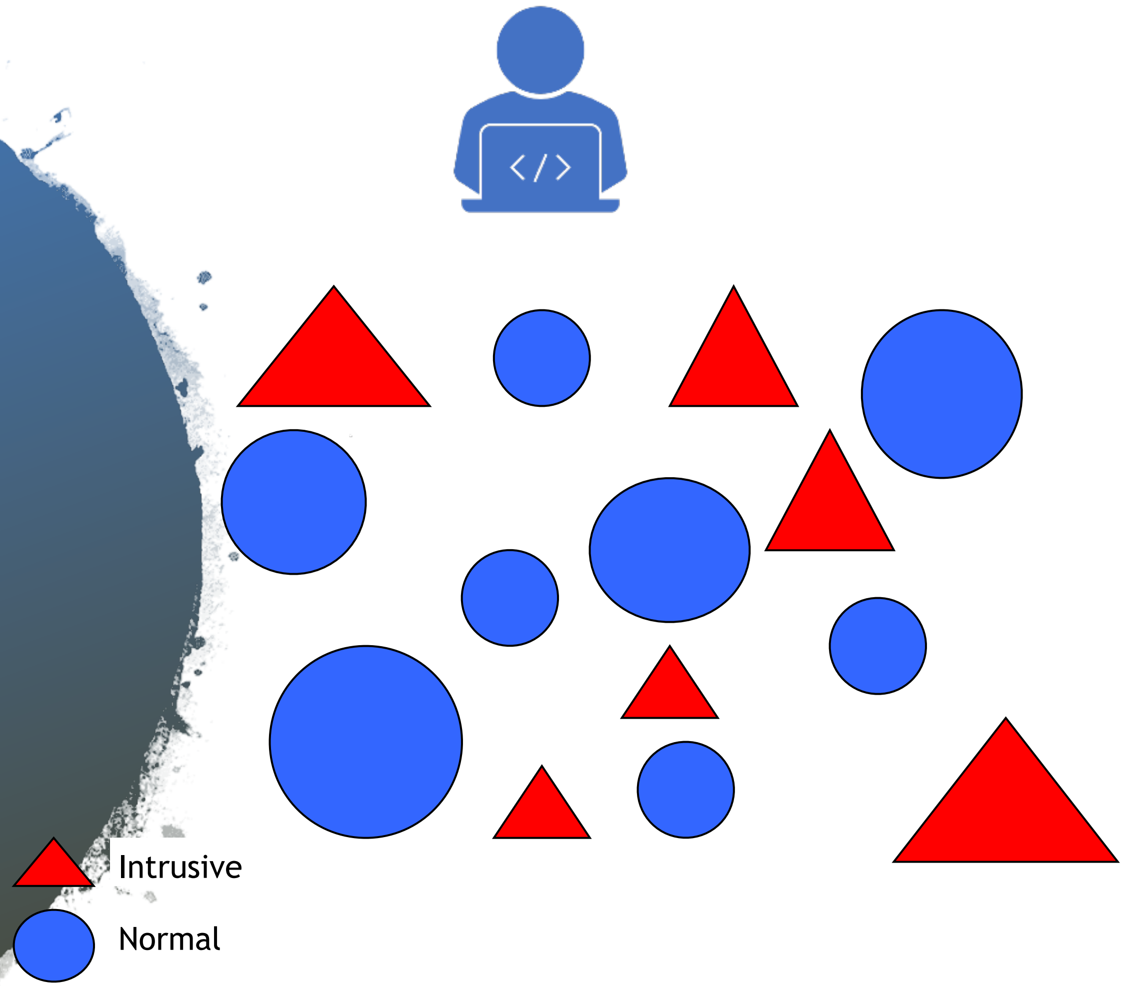
What Are Computer Attacks?

- 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

Comparison

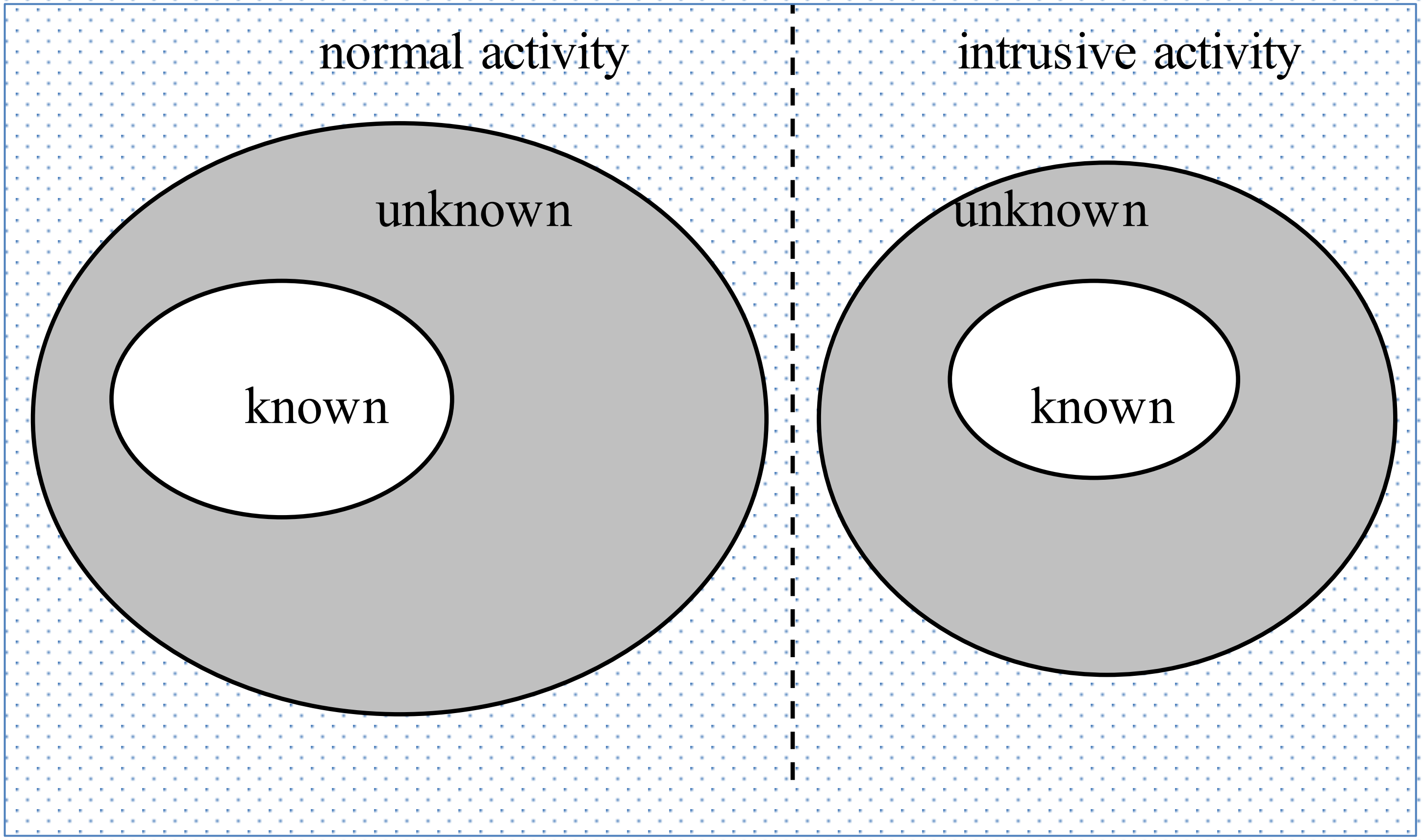
- Systems under attack do not meet at least one of these.
  - Non-privileged user may not normally acquire privilege (violates #1)
  - Attacker may break in using sequence of commands that violate security policy (violates #2)
  - Attacker may cause program to act in ways that violate program's specification (#3)

“Imaginary” Computer Activities



Denning's Model

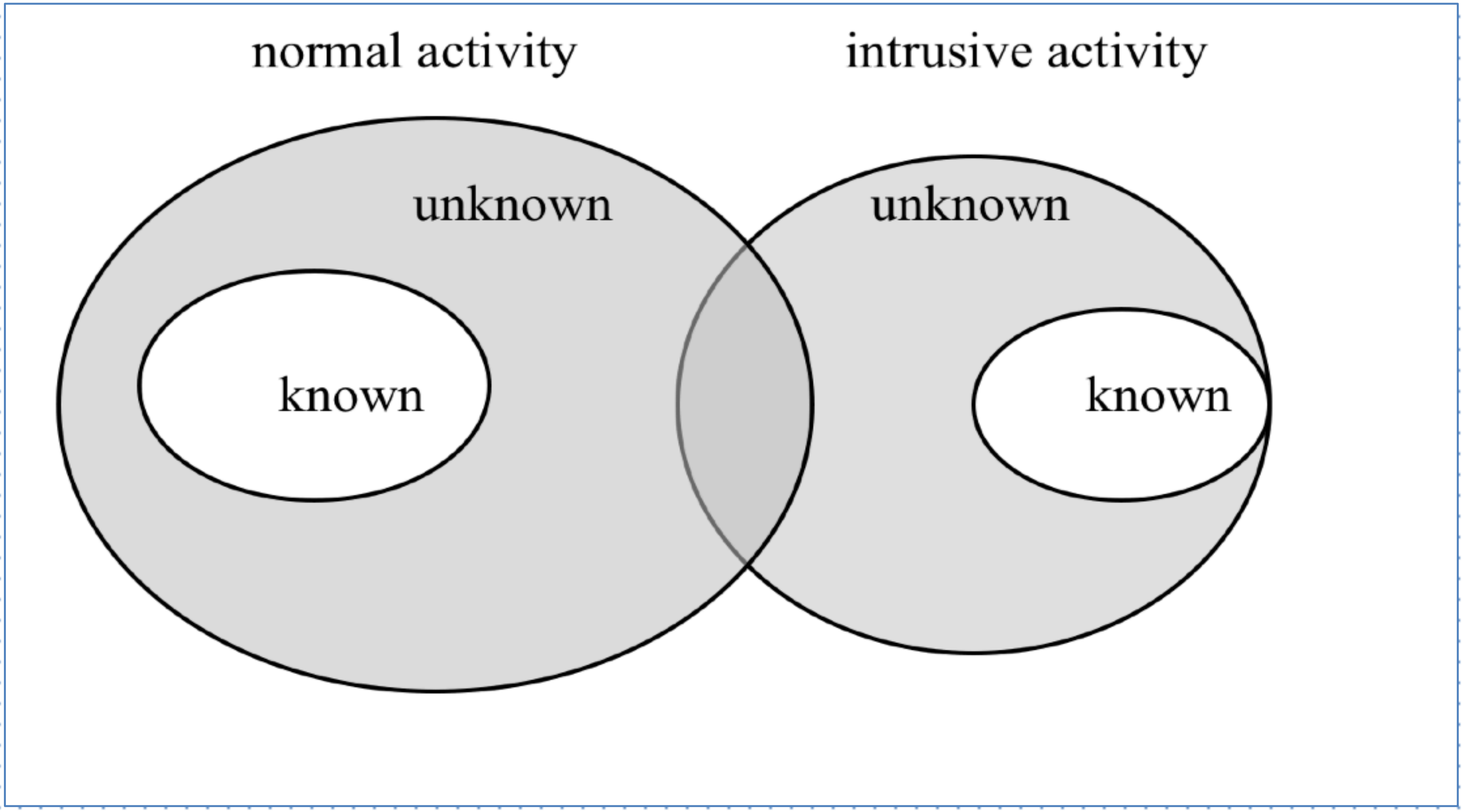
- Hypothesis: exploiting vulnerabilities requires abnormal use of normal commands or instructions
  - Includes deviation from usual actions
  - Includes execution of actions leading to break-ins
  - Includes actions inconsistent with specifications of privileged programs



Un/known normal and intrusive activities shown in a Venn diagram (Note there is no overlap in this view)

Assumptions

- Denning's initial assertion was that the region of "misuse" activity falls far enough outside the region of "normal" activity .
- The misuse detection proponents assert that the intersection is quite large.
- *Which is true?*



Un/known normal and intrusive activities in a computer system shown in a different Venn diagram

- Is this a more realistic view?

Outline

- Basic Concepts of Analysis and Detection
- What are attacks?
- Misuse Detection vs. Anomaly Detection
- Hybrid Architecture and Others
- Additional Issues

Classification of Detection Models

Anomaly Detection ("profile" based) builds up the normal profile of a subject and classifies the activities as attacks if they deviate significantly from the normal profile.

Misuse Detection ("signature" based) recognizes the patterns of intrusive activities (often with the help of normal activities) in training data. Then in detection this approach matches incoming data with these signatures.

Training

01

Collect and/or generate event information

- Attack activity vs. normal activity (labeled)

02

Preprocess the information

- Formatting, feature selection

03

Build a classification model

- Rules/patterns vs. statistical profile

04

Populate it with event data

- Instantiated to the specific system

05

Store the model in a knowledge base

IDES Measures

Figure 4.3 IDES Measure Categories and Examples

	Ordinal (Continuous)	Categorical (Discrete)
Binary	CPU time used Number of audit records produced	Whether a directory was used Whether a file was accessed Whether audit records indicated use for day/week/month
Linear		# of times each command was used # of system-related errors # of login failures in last hour # of audit events recorded # of files modified

Four classes of statistical measures:

- Intensity measures: e.g., number of audit records per time unit
- Categorical measures: names of remote hosts used
- Counting measures: CPU time used
- Audit record distribution measure

IDES /NIDES Training

- For each measure, obtain
  - A Q value to reflect the recent behavior (e.g., number of audit records in the recent past)
  - A relative frequency distribution of Q values as the norm profile of Q values from Q values collected over a long term:
    P1 = 1% of Q values in the range of 0-10 audit records
    P2 = 7% of Q values in the range of 11-20 audit records
    P3 = 35% of Q values in the range of 21-40 audit records
    P4 = 18% of Q values in the range of 41-80 audit records
    P5 = 28% of Q values in the range of 81-160 audit record
    P6 = 11% of Q values in the range of 160-320 audit records

Detection

01
- Monitor and input new event record

02
- Preprocessing data
  - Assembling the sessions

03
- Compare the event record to the knowledge base
  - Pattern matching vs. distance calculation

04
- Generate a response
  - A report, a further action, etc.

IDES /NIDES Detection

- If the Q value falls in the i^{th} interval, $TPROB_i$ as the sum of P_i and other smaller P_j's is calculated:
  - Q=200, $TPROB_6 = P_6 + P_2 + P_1 = 0.11+0.07+0.01=0.19$
- S_i, such that $P(|N(0,1)| \geq S_i) = TPROB_i$, producing a larger S value for a smaller P_i, that is, the less frequently a recent Q value, the larger the S value, the more likely an anomaly (the inverse-proportional relationship of P and S):
  - $TPROB_{6,}$ S_6= 0.84; Q=30, TPROB=1.0, S=0
- Not robust due to the assumption of normally distributed data,
  - e.g., P_1=4.9%, P_2=5%, …, P_9=5%, P_{10}=5.1%; P_1 and P_{10} produce S_1=2.25, and S_{10}=0

Testing

Similar to detection

Testing data labeled as normal/intrusive

Comparison of the ground truth and the prediction by the trained model

- True and false detection (positive)
- Find the best setting (e.g., model parameters and control threshold…)

Feedback and Refinement

Misuse detection

- Update of signature databases
- Management of state-retention in the event horizon
- Memory management for orphan session records

Anomaly detection

- Update the historic profiles

Misuse Detection

Based on characteristics of intrusive activities

Specific commands (e.g., su), IP addresses, event sequences

Manual encoding of intrusion signatures

State transition machine

Colored Petri-net

Rules

Automatic learning of intrusion signatures

Existing data mining techniques: decision trees, association rules, etc.

Existing clustering techniques: hierarchical clustering, K-means, etc.

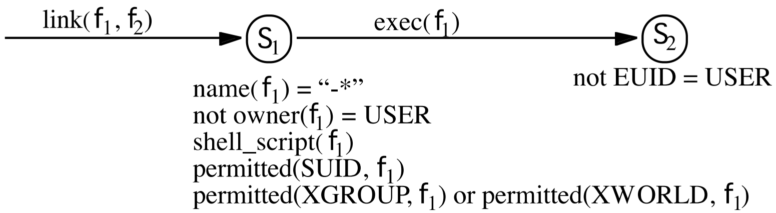
Problems: scalability, incremental updating, unknown number of clusters

STAT

- Analyzes state transitions
  - Need keep only data relevant to security
  - Example: look at process gaining *root* privileges; how did it get them?
- Example: attack giving setuid to *root* shell

```
ln target ./–s
–s
```

State Transition Diagram



- Conditions met when system enters states s_1 and s_2; USER is effective UID of process
- Note final postcondition is USER is no longer effective UID; usually done with new EUID of 0 (*root*) but works with any EUID

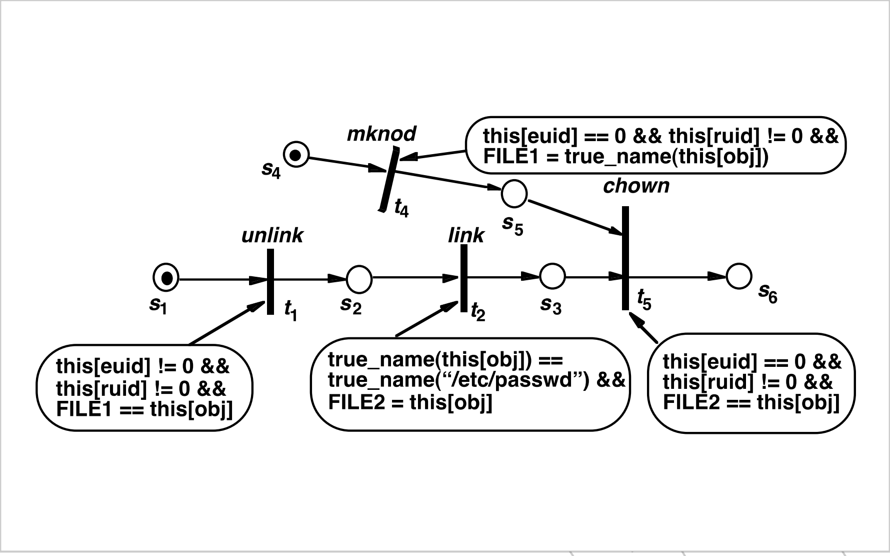
How Inference Engine Works

- Constructs series of state table entries corresponding to transitions
- Example: rule base has single rule above
  - Initial table has 1 row, 2 columns (corresponding to s1 and s2)
  - Transition moves system into s1
  - Engine adds second row, with “X” in first column as in state s1
  - Transition moves system into s2
  - Rule fires as in compromised transition
    - Does not clear row until conditions of that state false

State Table

	s_1	s_2
1		
2	X	

now in s_1



Another Example: IDIOT

Markov Chain

- Markov chain model of normal activities
  - Event transition: (X(1), …, X(t))
  - Training: Attack sequences (and normal sequences)
  - Markov chain model
    - Training:

$$Q = \begin{bmatrix} q_1 & q_2 & \boxed{?} & q_s \end{bmatrix} \qquad P = \begin{bmatrix} p_{11} & p_{12} & \boxed{?} & p_{1s} \\ p_{21} & p_{22} & \boxed{?} & p_{2s} \\ \boxed{?} & \boxed{?} & \boxed{?} & \boxed{?} \\ p_{s1} & p_{s2} & \boxed{?} & p_{ss} \end{bmatrix}$$

$$q_i = \frac{N_i}{N} \qquad p_{ij} = \frac{N_{ij}}{N_{i.}}$$

    - Testing:

$$P(X_1, \boxed{?}, X_T) = q_{x_1} \prod_{t=2}^{T} P_{X_{t-1} X_t}$$

Anomaly Detection

- Use characteristics (profile) of normal activities, and compare the recent past versus long term
- Some examples
    - Prediction-based profiling (artificial neural network, regression)
    - String-based profiling (computer immunology)
    - Probability-based profiling (Markov chain, Bayesian network, Hidden Markov Models)
    - Statistics-based profiling (IDES/NIDES, Hotelling's T2, chi-square distance test, Canberra distance test)

SPC techniques

- Existing Multivariate Statistical Process Control (MSPC) techniques
  - Hotelling's T^2 test:

$$T^2 = \left(X - \overline{X}\right)' S^{-1} \left(X - \overline{X}\right)$$

- Problems: scalability and difficulty in computing the variance-covariance matrix and its inverse

Scalable SPC techniques

- Chi-squared distance:

$$X^2 = \sum_{i=1}^{284} \frac{\left(X_i - \overline{X_i}\right)^2}{\overline{X_i}}$$

- Canberra distance:

$$C = \sum_{i=1}^{284} \frac{\left|X_i - \overline{X_i}\right|}{X_i + \overline{X_i},}$$

Again: Markov Chain

- Markov chain model of normal activities
  - Event transition: (X(1), …, X(t))
  - Training: Normal sequences
  - Markov chain model
    - Training:

$$Q = \begin{bmatrix} q_1 & q_2 & \boxed{?} & q_s \end{bmatrix}$$

$$q_i = \frac{N_i}{N}$$

$$P = \begin{bmatrix} p_{11} & p_{12} & \boxed{?} & p_{1s} \\ p_{21} & p_{22} & \boxed{?} & p_{2s} \\ \boxed{?} & \boxed{?} & \boxed{?} & \boxed{?} \\ p_{s1} & p_{s2} & \boxed{?} & p_{ss} \end{bmatrix}$$

$$p_{ij} = \frac{N_{ij}}{N_{i.}}$$

    - Testing:

$$P(X_1, \boxed{?}, X_T) = q_{x1} \prod_{t=2}^{T} P_{X_{t-1} X_t}$$

Outline

- Basic Concepts of Analysis and Detection
- What are attacks?
- Misuse Detection vs. Anomaly Detection
- Hybrid Architecture and Others
- Additional Issues

Other Detection Models?

Anomaly detection

What is usual, is known

What is unusual, is bad

Misuse detection

What is bad is known

Specification-based detection

We know what is good

What is not good is bad

Specification Modeling

- Determines whether execution of sequence of instructions violates specification
- Only need to check programs that alter protection state of system
- System traces, or sequences of events t_1, … t_i, t_{i+1}, …, are basis of this
  - Event t_i occurs at time $C(t_i)$
  - Events in a system trace are totally ordered

System Traces

- Notion of *subtrace* (subsequence of a trace) allows you to handle threads of a process, process of a system
- Notion of *merge of traces U*, *V* when trace *U* and trace *V* merged into single trace
- *Filter p* maps trace T to subtrace T' such that, for all events $t_i \in T'$, $p(t_i)$ is true

Examples

- Subject *S* composed of processes *p*, *q*, *r*, with traces T_p, T_q, T_r has $T_s = T_p \oplus T_q \oplus T_r$
- Filtering function: apply to system trace
  - On process, program, host, user as 4-tuple

  `< ANY, emacs, ANY, bishop >`

  lists events with program "emacs", user "bishop"

  `< ANY, ANY, nobhill, ANY >`

  list events on host "nobhill"

Example: Apply to *rdist*

- Ko, Levitt, Ruschitzka defined PE-grammar (parallel environment grammars) to describe accepted behavior of program.
- *rdist* creates temp file, copies contents into it, changes protection mask, owner of it, copies it into place.
  - Attack: during copy, delete temp file and place symbolic link with same name as temp file
  - *rdist* changes mode, ownership to that of program

Relevant Parts of Specification

7. SE: <rdist>
8. <rdist> -> <valid_op> <rdist> |.
9. <valid_op> -> open_r_worldread

   …

   | chown

   { if !(Created(F) and M.newownerid = U)

   then violation(); fi; }

   …
10. END

- chown of symlink violates this rule as M.newownerid ≠ U (owner of file symlink points to is not owner of file *rdist* is distributing)

Comparison of Detection Models

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.

Challenges to Detection Models

Misuse detection

- Limited by available signatures
- Can’t detect “new” attacks
- Must be updated frequently

Anomaly detection

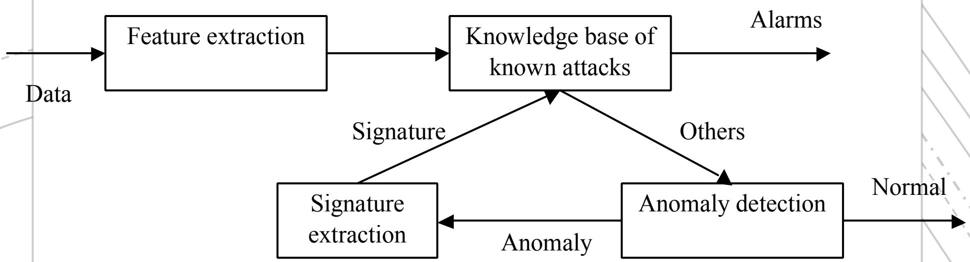
- Requires representative normal data or attack-free data
- May classify new normal activities as attacks

Specification modeling

- Still in infancy
- Extra efforts needed to locate and analyze many programs

Hybrid Architecture of Intrusion Detection

- Different categories of detection models complement each other in terms of detection ability of new types of attack and false alarm rate.
- It is natural to design a hybrid intrusion detection system that applies these two techniques together to improve the overall performance.
- This design is very similar to the so-called MINDS system developed at the University of Minnesota (Ertoz et al, 2004).



The structure of a hybrid intrusion detection system

Outline

- Basic Concepts of Analysis and Detection
- What are attacks?
- Misuse Detection vs. Anomaly Detection
- Hybrid Architecture and Others
- Additional Issues

IDS Operation Timing

Batch/interval/Off-line

- Analysis is done on bulk data (files)/logs
- Analysis is periodic
- Examples: ASIM, NADIR, Stalker, Tripwire

Real-time

- Monitor the system continuously
- Report suspicious activities as soon as possible
- Results can be used to take timely action
- Examples: AAFID, Bro, CMDS, CSM, DIDS, EMERALD, GRIDS, INBOUNDS, MIDAS, NIDES

Challenge to Operation Timing

Time taken and processing cost

- How long is the data available for analysis?
- How many signatures/patterns can be checked?
- Is there time to react?

Violations within idle interval

- A file modification between Tripwire runs

IDS Control and Architecture

Centralized	Hierarchical	Agent-based/ distributed
• Data from one or multiple hosts • Central repository for analysis • Example: Tripwire (host) and SNORT (network)	• Layers of analysis • Example: EMERALD	• Distributed collection using agents or sensors • Distributed analysis • Alerts can be sent to central collection point • Example: AAFID (network)

Challenge to Control Structure

Centralized

- Sufficient processing resources
- Communication load
- Protection from attack

Agent-based

- Subversion of agents
- Secure communication
- Efficiency of agents

Monitoring Strategy

Host based

- Collect data from a single host
- Examples: MIDAS, Tripwire

Multi-host based

- Analyze data from multiple hosts
- Examples: AAFID, DIDS, NIDES, Stalker

Network based

- Examine network traffic
- Examples: Bro, CyberCop, EMERALD, GRIDS, NADIR

Challenge to Monitoring Strategy

Host-based	Multi-host based	Network-based
No global knowledge or context information Overhead to host being monitored Security of the host Recovery options are limited	Larger volume data	Network data rates are very high Encryption of network traffic is becoming more popular Difficult to insure that network IDS sees the same data as the end hosts