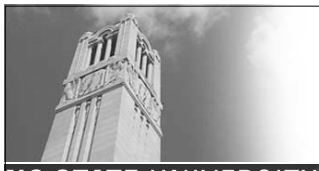


NC STATE UNIVERSITY Computer Science

CSC 774 Network Security

Topic 8.1 Introduction to Intrusion Detection

Outline

- The circle of prevention, detection & response
- Intrusion and intrusion detection
- Methodologies for intrusion detection
  - Anomaly detection
  - Misuse detection
- Selected intrusion detection methods
- Evaluation of intrusion detection systems (IDSs)

Approaches to Protecting Information Systems

- Prevention
  - E.g., Encryption, Authentication, Access control
  - Cryptography underlies most of prevention-based approaches.
- Detection & Response

Prevention

- The best "medicine."
  - System and protocol designs contain no security vulnerabilities.
  - Implementations verifiably secure with respect to the design spec.
  - No bugs in either hardware or software.
  - All systems are configured to avoid any security holes.
  - Everyone practices secure networking...
- Effective prevention remains a dream
  - "If you think small and simple, you are doomed to fail" - all the attacker needs is a "hole"

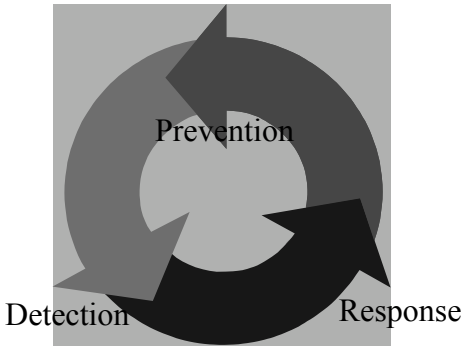
Detection

- The first step to protection when a security breach happens
  - Breaches due to hardware and software failures (faults and bugs)
  - Breaches due to user error (system administrator and end user etc.)
  - Breaches caused by malicious attackers
- When there is no security breach
  - Defense in depth

Response

- Yes, we've got to do something!
  - Source isolation
  - Intrusion containment
  - Damage control
  - System reconstitution
  - Intention and trend analysis
  - Security assessment
  - Detection & response reconfiguration
  - System hardening

Circle of Security Continues



Intrusion and Intrusion Detection

- Intrusion
  - Intrusions in an information system are the activities that violate the security policy of the system
- Intrusion detection
  - The process of identifying intrusions

Intrusion Taxonomy (incomplete)

- Masquerading
- Subsequent misuse (e.g. plan a bug)
- Control bypass
- Active resource misuse (change)
- Passive resource misuse (read)
- Denial-of-service
- Misuse via inaction

Intrusion versus Faults/Failures

- Natural faults
  - Hardware failure/software bugs
  - Different fault models from fault tolerance and reliability discipline
  - Fault detection approach applies
- Faults by (malicious) design
  - Still subject to the same fault model
  - Natural fault detection mechanism will detect these as special cases (intentionally caused)

Known versus Unknown

- Known intrusions
  - Well understood attacks
  - Known indication/signatures - easier to detect
    - For example, "copy of /etc/passwd by an unprivileged user"
- Unknown (newly invented) intrusions
  - Usually very difficult to detect using existing IDSs
  - Can you detect any at all?

Classification of Intrusion Detection Techniques (Systems)

- Anomaly detection
  - Based on the normal behavior of a subject (e.g., a user)
  - Any action that significantly deviates from the normal behavior is considered intrusive.
- Misuse detection
  - Catch intrusions in terms of the characteristics of known intrusions or system vulnerabilities
  - Any action that conforms to the pattern of a known attack or vulnerability is considered intrusive.

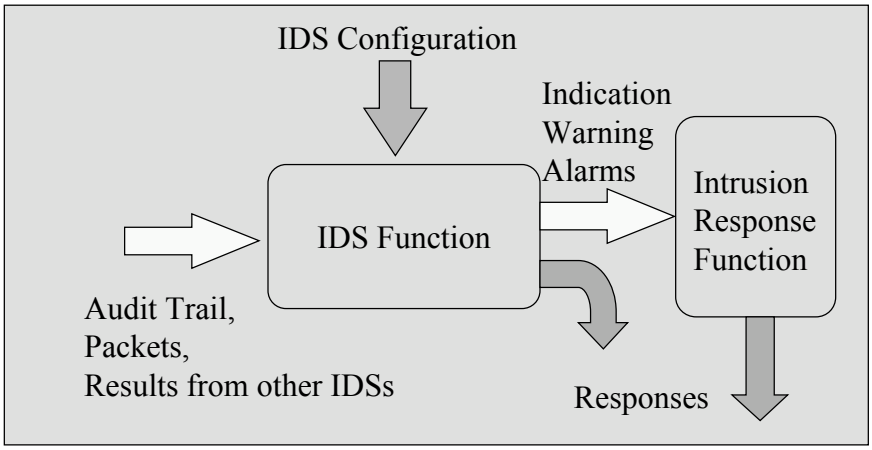
Another Categorization

- Based on the data source
  - Host-based intrusion detection
    - Intrusion detection by analyzing host audit trail
    - Example: NIDES
  - Network-based intrusion detection
    - Intrusion detection by analyzing network traffic
    - Example: Snort
  - Distributed intrusion detection
    - Intrusion detection by analyzing audit trails from multiple hosts and possibly network traffic
    - Example: EMERALD

Elements of Intrusion Detection

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

Intrusion Detection System



Each function can be implemented in a
a centralized or distributed fashion

Network-Based IDSs

- Deploying special sensors at strategic locations
  - E.G., Packet sniffing via tcpdump at routers
- Inspecting network traffic
  - Watch for violations of protocols and unusual connection patterns
- Monitoring user activities
  - Look into the data portions of the packets for malicious command sequences
- May be easily defeated by encryption
  - Data portions and some header information can be encrypted

Host-based IDSs

- Using OS auditing mechanisms
  - E.G., BSM on Solaris: logs all direct or indirect events generated by a user
  - strace for system calls made by a program
- Monitoring user activities
  - E.G., Analyze shell commands
- Monitoring executions of system programs
  - E.G., Analyze system calls made by sendmail

Selected Intrusion Detection Methods

- Anomaly detection
  - Statistical methods
    - NIDES/STAT
  - Machine learning and data mining methods
    - Instance based learning
  - Specification-based method
- Misuse detection
  - Rule-based languages
    - Snort
  - State transition analysis Toolkit (STAT)
  - Automatically building misuse detection models

NIDES/STAT: A Statistical Method

- Normal Profile
  - Based on a number of measures M_1 through M_n
  - Examples: # files opened, CPU usage, memory usage
- Abnormal values
  - S_1, S_2, …, S_n represent the abnormality values of M_1 through M_n
    - Differences between the observed and the profiled values
  - Overall statistics $T^2 = S_1^2 + S_2^2 + \ldots + S_n^2$.
  - Abnormal if $T^2 >$ threshold.

NIDES/STAT (Cont'd)

- Profile update
  - NIDES/STAT multiplies the frequency table in each profile by an exponential decay factor before incorporating the new audit data.
  - Allow automatic update of profile
  - An attacker may gradually "train" the profile to consider his/her intrusive activities as normal behavior

Specification-Based Approach

- Basic idea
  - Specify the "intended" behavior of the programs
  - Raise alarm on deviation from the spec
  - Catch both known and unknown attacks
- Detection examples
  - Privileged program vulnerability exploits
  - Race conditions
  - Security violation caused by synchronization in a distributed system

Ways to Get Specifications

- Manual specification
  - Time consuming and error prone
- Learning from program executions under normal situations
  - May learn unknown intrusive behaviors as normal ones
- Reuse existing specifications
  - TCP/IP state machines
- Static analysis
  - Derive an abstract representation of the program

STAT Approach

- State transition analysis tool
- USTAT: Unix, real-time version
- NetSTAT:
- Intrusion/penetration detection tool
  - You must know the attacks first
  - Offer state diagrams to catch each of the attacks
  - State diagram tracks and detects known attacks

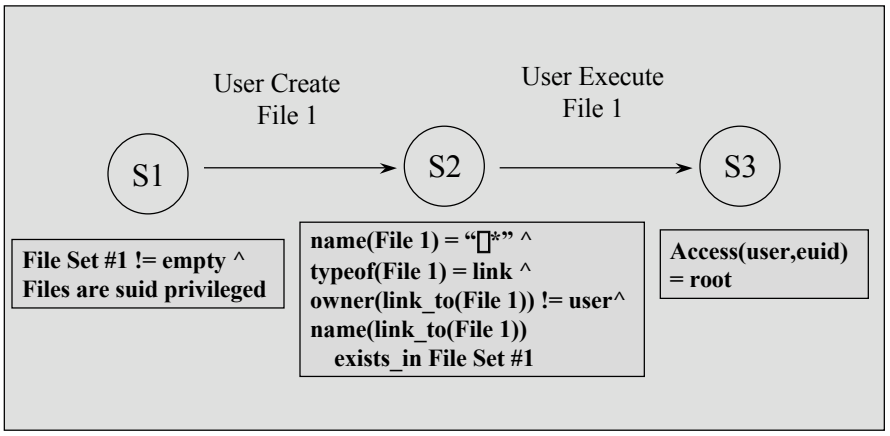
A Penetration Scenario

- Applicable to SunOS 4.1.1.
- *target* is a setuid shell script with #!/bin/sh mechanism and is owned by root.
- The attacker executes the following.

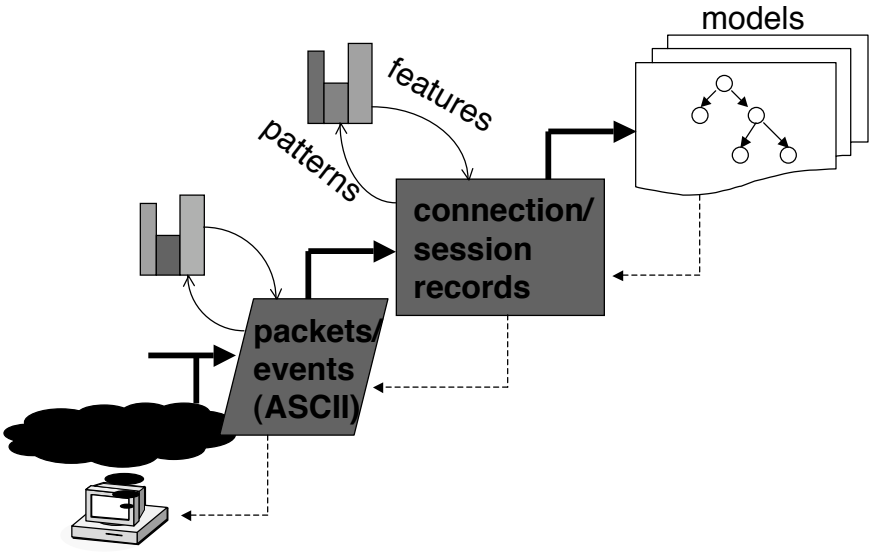
  %ln target –x

  %–x
- Insight: executing –x starts an interactive shell with root privileges

STAT Diagram for the Penetration



A Data Mining Process of Building Misuse Detection Models



Data Mining

- Relevant data mining algorithms for ID
  - Classification: maps a data item to a category (e.g., normal or intrusion)
    - Rule learner
  - Link analysis: determines relations between attributes (system features)
    - Association rules
  - Sequence analysis: finds sequential patterns
    - Frequent episodes

Classifiers As ID Models

- Classification rule learner:
  - Use the most distinguishing and concise attribute/value tests for each class label.
- Example rule-set:
  - if (wrong_fragment ≥ 1 AND protocol_type = icmp) then "pod."
  - else if (protocol = icmp_echo_request AND host_count ≥ 3 AND srv_count ≥ 3) then "smurf."
  - ...
  - else normal.

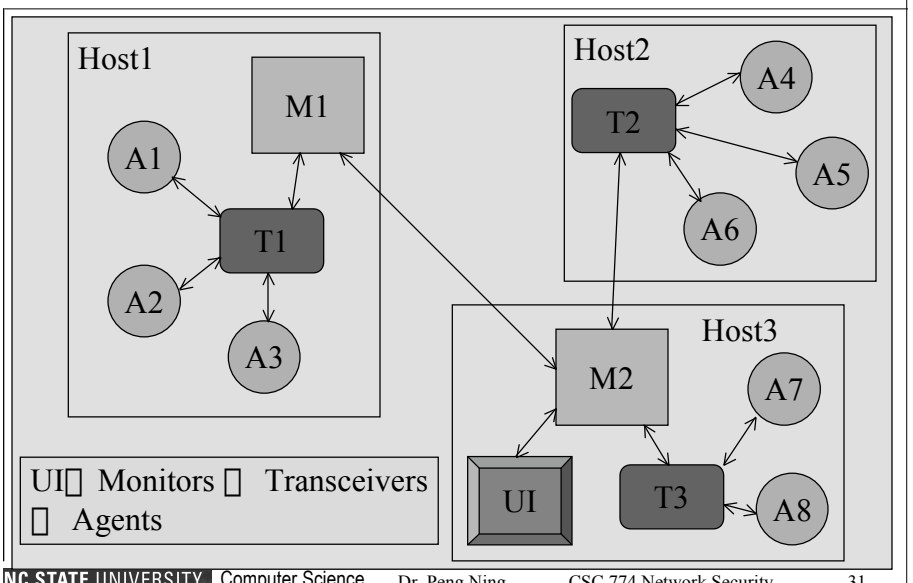
Classifiers As EFFECTIVE ID Models

- Need features with high information gain, i.e., reduction in entropy (a measure of data "impurity")
  - temporal and statistical features for ID
- Solution:
  - Mine frequent sequential patterns
  - Identify "intrusion-only" patterns and construct features accordingly
    - The constructed features have high information gain

AAFID: A Distributed Intrusion Detection Architecture

- AAFID: Autonomous Agents for Intrusion Detection
- Goals
  - Continually running
  - Fault tolerant
  - Resist subversion
  - Minimal overhead
  - Amenable to policy
  - Adaptive to system and user behavior change

AAFID Architecture



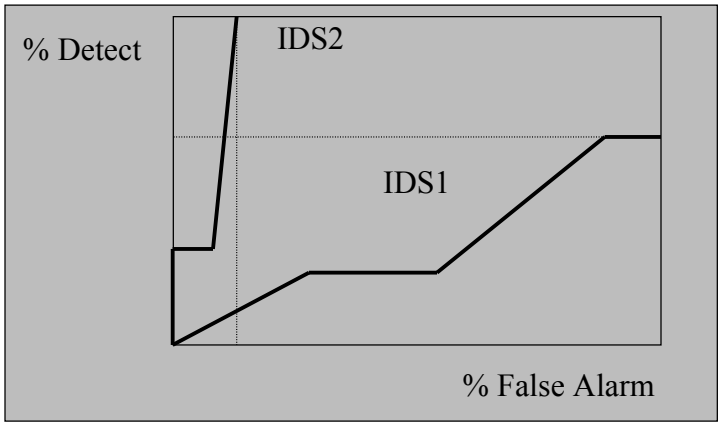
AAFID (Cont'd)

- Agent - an independently running entity that monitors specific aspects of a host and reports abnormal/interesting events
- Transceiver
  - Start and stop agents
  - Keep track of the agent status on the host
  - Receive and process reports from agents
  - Distribute the results to monitors or agents
- Monitors
  - Controls entities on multiple hosts
  - Process (correlate) results from multiple transceivers
  - Interaction with other monitors and UI

Evaluation of IDS

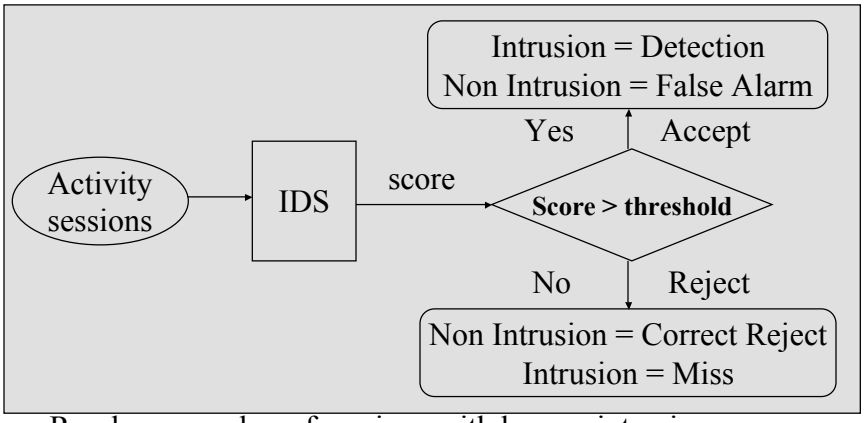
- Type I error: (false negative)
  - Intrusive but not being detected
- Type II error: (false positive)
  - Not intrusive but being detected as intrusive
- Evaluation:
  - How to measure?
  - ROC - receiver operating characteristics curve analysis - detection rate vs. False alarm rate
  - What else? Efficiency? "Cost?"

Example ROC Curves



- Ideal system should have 100% detection rate with 0% false alarm

Drawing ROC Curves



- Run large number of sessions with known intrusion distribution
- Vary threshold to obtain false alarms & misses to create ROC curves

Further Readings

- Axelsson, S. (1999). *Research in intrusion-detection systems: A survey*. *Technical report TR 98-17*. Göteborg, Sweden: Department of Computer Engineering, Chalmers University of Technology.
- Ko, C., Ruschitzka, M., & K. Levitt (1997). Execution monitoring of security-critical programs in distributed systems: a apecification-based approach. In G. Dinolt & P. Karger (Eds.), *Proceedings of 1997 IEEE symposium of security and privacy* (pp. 175-187), IEEE Computer Socitey, Los Alamitos, CA.
- Lee, W., & Stolfo, S.J. (2000). A framework for constructing features and models for intrusion detection systems. *ACM Transactions on Information and System Security*, *3* (4) (pp. 227-261).
- Spafford, E.H., & Zamboni, D. (2000). Intrusion detection using autonomous agents. *Computer Networks 34*, 547--570.
- Vigna, G., & Kemmerer, R.A. (1999). NetSTAT: A network-based intrusion detection system. *Journal of Computer Security*, *7* (1), 37--71.
- Wagner, D., & Dean, D. (2001). Intrusion detection via static analysis. In R. Needham & M. Abadi (Eds), *Proceedings of 2001 IEEE symposium on security and privacy* (pp. 156--168), IEEE Computer Socitey, Los Alamitos, CA.
- Sekar, R., Bendre, M., Dhurjati, D., & Bollineni, P. (2001). A fast automaton-based method for detecting anomalous program behaviors. In R. Needham & M. Abadi (Eds), *Proceedings of 2001 IEEE symposium on security and privacy* (pp. 144--155), IEEE Computer Socitey, Los Alamitos, CA.