Developing Data Mining Techniques for Intrusion Detection: A Progress Report

Wenke Lee

Computer Science Department,

North Carolina State University

Outline

- Intrusion detection: promises and challenges
- A development process using data mining
- Cost-sensitive analysis and modeling
- Anomaly detection

Building a Secure Network

Vulnerabilities

↓

Prevention

↓

Intrusion Detection

↓

Response/ Tolerance

Intrusion Detection

- Primary assumptions:
  - System activities are observable
  - Normal and intrusive activities have distinct evidence
- Main techniques:
  - Misuse detection: patterns of well-known attacks
  - Anomaly detection: deviation from normal usage

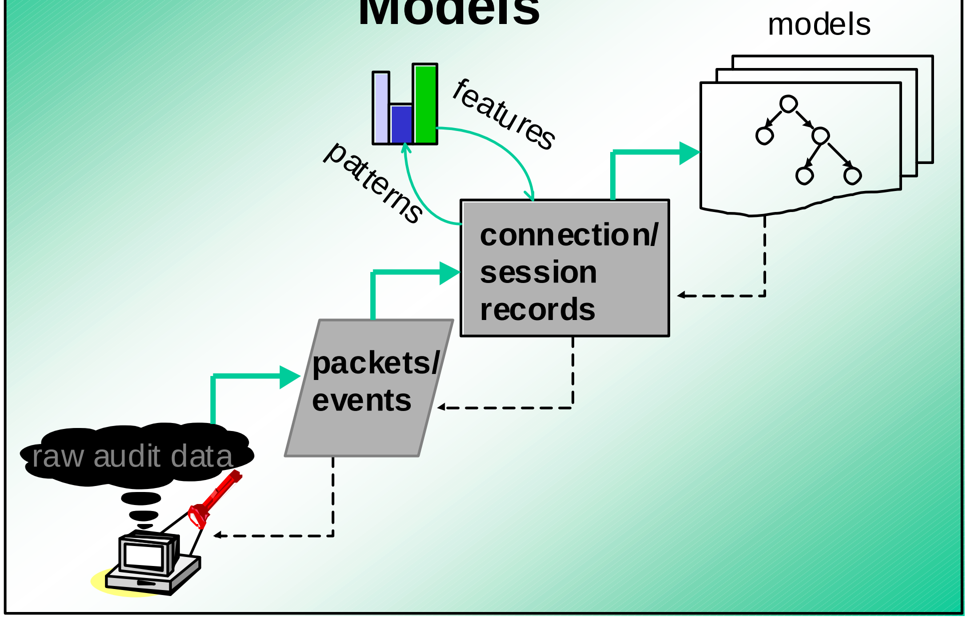
The State of Current ID Techniques

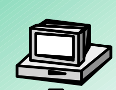
- Poor effectiveness:
  - Marginal true positive rate:
    - Signatures not adaptive to new network environments and attack variations
  - High false positive rate:
    - Especially for anomaly detection
- Poor theoretical foundations and development methodology
  - Pure knowledge engineering.
  - But the networking environment too complicated.

DM for Building ID Models

- Motivation:
  - A systematic IDS development toolkit.
- Approach:
  - Mine activity patterns from audit data;
  - Identify “intrusion patterns” and construct features;
  - Build classifiers as ID models.
- Results:
  - One of the best performing systems in 1998 DARPA Evaluation.

The DM Process of Building ID Models





tcpdump **packet data**

```
10:35:41.5 A > B : . 512:1024(512) ack 1 win 9216
10:35:42.2 C > D: . ack 1073 win 16384
10:35:45.6 E > F: . ack 2650 win 16225
...
```

connection records

time	dur	src	dst	bytes	srv	flag	…
10:35:39.1	5.2	A	B	42	http	SF	…
10:35:40.4	20.5	C	D	22	user	REJ	…
10:35:41.2	10.2	E	F	1036	ftp	SF	…
…	…	…	…	…	...	…	…

Data Mining for ID

- Relevant data mining algorithms:
  - – 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/uncertainty")
  - temporal and statistical features for ID
- Our approach:
  - Mine frequent sequential patterns
  - Identify "intrusion-only" patterns and construct features accordingly
    - The constructed features have high information gain

Mining Audit Data

- Basic (standard) algorithms:
  - Association rules: intra-audit record patterns
  - Frequent episodes: inter-audit record patterns
  - Need both
- Need to efficiently compute only the "relevant" patterns:
  - Utilize schema-level information
  - The "interestingness" of a pattern = whether it contains "important" attributes

Association Rules

- Motivation:
  - Correlation among system features
  - $X \rightarrow Y$ [c, s]
    - c: confidence
    - S: support
- Example from shell commands:
  - *Mail* $\rightarrow$ *am, hostA* [0.3, 0.1]

Frequent Episodes

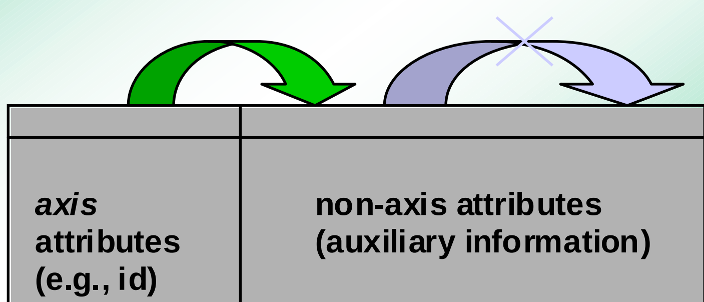
- Motivation:
  - Sequential information (system activities)
  - X, $Y \rightarrow Z$ [c, s, w]
    - X,Y, and Z are in different records
    - these records are within w
- Example from shell commands:
  - (*vi, C, am*) $\rightarrow$ (*gcc, C, am*) [0.6,0.2,5]

Extensions to Data Mining Algorithms

- Designating the “important” attributes to compute “relevant” patterns
  – *axis* attribute(s)
  – *reference* attribute(s)
- Uncovering low frequency but important patterns
  – *level-wise* approximate mining
  – mining with *relative support*

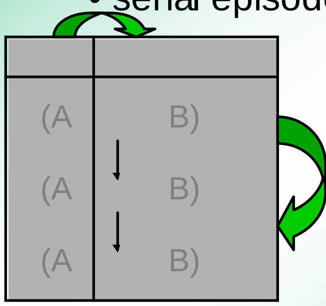
Axis Attribute(s)

– *axis* attribute(s) as an “item constraint”:
- the most important attribute, e.g., *service*
- an itemset must contain *axis* attribute values



Axis Attribute(s) (Continued)

– Compute sequential patterns in two phases:
  - associations using the axis attribute(s)
  - serial episodes from associations



Example (*service* is the axis attribute):

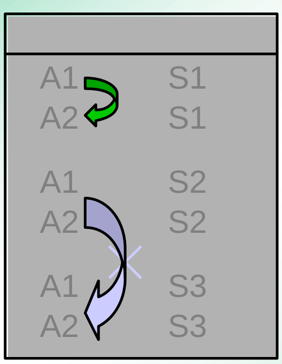
(*service* = telnet, *src_bytes* = 200, *dst_bytes* = 300, *flag* = SF), (*service* = smtp, *flag* = SF) $\rightarrow$ (*service* = telnet, *src_bytes* = 200)

Reference Attribute(s)

the “subject” of a sequence of related “actions”

e.g., connections to the same destination host:

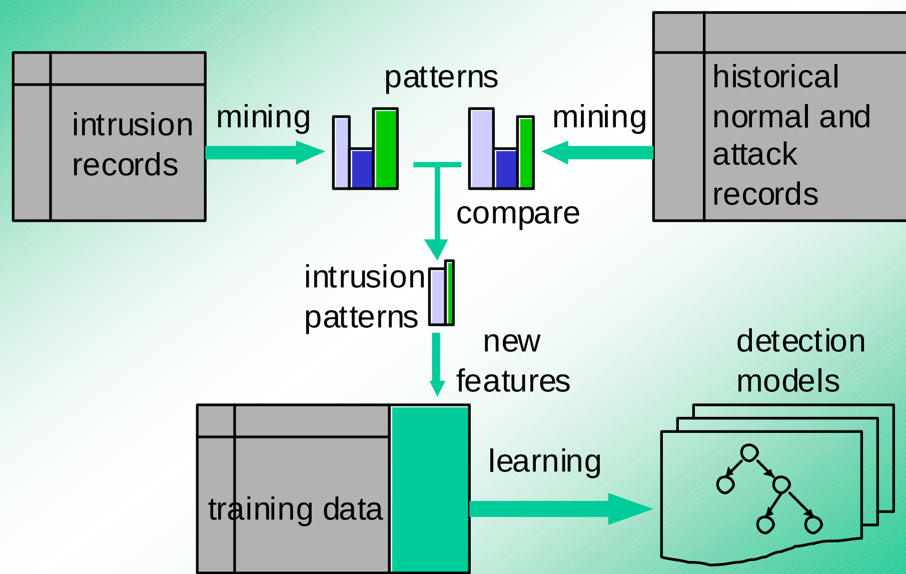
(service=http, flag=S0), (service=http, flag=S0) $\rightarrow$ (service=http, flag=S0)



reference attribute(s) as an item constraint:

records of an episode must have the same *reference* attribute value

Feature Construction from Patterns



Feature Construction from Patterns

- Parse an “intrusion-only” pattern
  - Identify the anatomy (in reference and axis attribute(s)) and invariant (in non-axis attribute(s)) of an attack;
  - Add features that use *count*, *percent*, and *average* operations on the attribute values in the pattern.

Feature Construction Example

- An example: “syn flood” patterns (dst_host is *reference* attribute):
  - (flag = S0, service = http), (flag = S0, service = http,) $\rightarrow$ (flag = S0, service = http) [0.6, 0.1, 2]
  - add features:
    - count the connections to the same *dst_host* in the past 2 seconds, and among these connections,
    - the percentage with the same *service*,
    - the percentage with S0

Theoretical Underpinnings

- “Intrusion-only” patterns identify how a set of intrusion records differ from normal records
- Features constructed from these patterns separate intrusion and normal records
  - i.e., they have high information gain
  - hence resulting classifiers more accurate

1998 DARPA ID Evaluation

- The data:
  - Total 38 attack types, in four categories:
    - DOS (denial-of-service), e.g., syn flood
    - Probing (gathering information), e.g., port scan
    - r2l (remote intruder illegally gaining access to local systems), e.g., guess password
    - u2r (user illegally gaining root privilege), e.g., buffer overflow
  - 40% of attack types are in test data only, i.e., “new” to intrusion detection systems
    - to evaluate how well the IDSs generalized

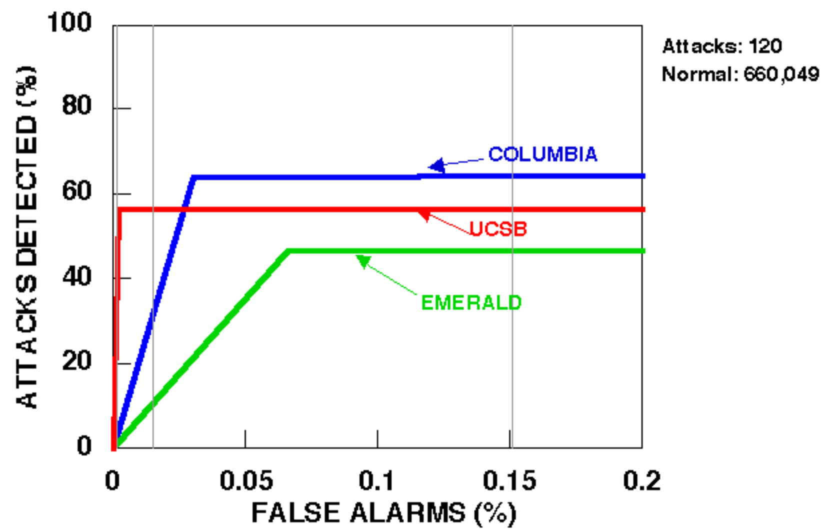
DARPA ID Evaluation (cont’d)

- Features:
  - “intrinsic” features:
    - protocol (service),
    - protocol type (tcp, udp, icmp, etc.)
    - duration of the connection,
    - flag (connection established and terminated properly, SYN error, rejected, etc.),
    - # of wrong fragments,
    - # of urgent packets,
    - whether the connection is from/to the same ip/port pair.

DARPA ID Evaluation (cont'd)

- Features constructed from mined patterns:
  - temporal and statistical "traffic" features that describe **connections** within a time window:
    - # of connections to the **same *destination host*** as the current connection in the past 2 seconds, and among these connections,
    - % of rejected connections,
    - % of connections with "SYN" errors,
    - % of different services,
    - % of connections that have the same service,
    - % of different (unique) services.

TCPDump Overall



MIT Lincoln Laboratory

Major Limitations

- Mainly misuse detection
- Requires labeled training data
  - not realistic for many environments
- Assumes fixed “session” definition, e.g., network connection
  - attacks can be extended and coordinated
- Need well engineered approach for real-time performance

The Need for Cost-sensitive ID

- High-volume automated attacks can overwhelm an IDS and its staff.
- Use cost-sensitive data mining algorithms to construct ID models that consider cost factors:
  - damage cost, response cost, operational cost, etc.
- Multiple specialized and light ID models can be dynamically activated/configured.
- Cost-effectiveness as the guiding principle and multi-model correlation as the architectural approach .

Cost Factors of IDSs

- Attack taxonomy: result/target/technique
- Development cost
- Damage cost (DCost)
  – The amount of damage when ID is not available or ineffective.
- Response cost (RCost)
  – The cost of acting upon an alarm of potential intrusion.
- Operational cost (OpCost)
  – The cost of processing and analyzing audit data ;
  – Mainly the computational costs of the features.

Cost Models of IDSs

- The total cost of an IDS over a set of events:
  - $CumulativeCost(E) = \sum_{e \in E} (CCost(e) + OpCost(e))$
- $CCost(e)$, the consequential cost, depends on prediction on event e

Consequential Cost (CCost)

- For event *e* :

Outcome	***CCost*(*e*)**	**Conditions**
Miss (*FN*)	*DCost*(*e*)	
False Alarm (*FP*)	*RCost*(*e'*)+*PCost*(*e*) 0	*DCost*(*e'*) ≥ *RCost*(*e'*) Otherwise
Hit (*TP*)	*RCost*(*e*)+*εDCost*(*e*) *DCost*(*e*)	*DCost*(*e*) ≥ *RCost*(*e*) Otherwise
Normal (*TN*)	0	
Misclassified Hit	*RCost*(*e'*)+*εDCost*(*e*) *DCost*(*e*)	*DCost*(*e'*) ≥ *RCost*(*e'*) Otherwise

Cost-sensitive Modeling: Objectives

- Reducing operational costs:
  - Use cheap features in ID models.
- Reducing consequential costs:
  - Do not respond to an intrusion if RCost > DCost.

Cost-sensitive Modeling: Approaches

- Reducing operational costs:
  - – A multiple-model approach:
    - Build multiple rule-sets, each with features of different cost levels;
    - Use cheaper rule-sets first, costlier ones later only for required accuracy.
  - – Feature-Cost-Sensitive Rule Induction:
    - Search heuristic considers information gain **AND** feature cost.

Cost-sensitive Modeling: Approaches (continued)

- Reducing consequential costs:
  - – MetaCost:
    - Purposely re-label intrusions with Rcost > DCost as normal.
  - – Post-Detection decision:
    - Action depends on comparison of RCost and DCost.

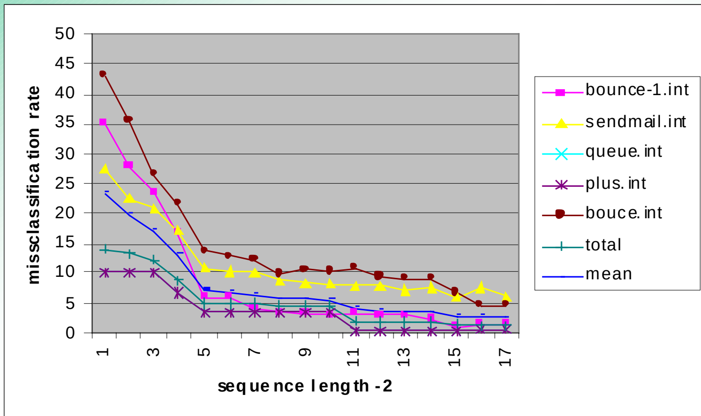
Anomaly Detection

- Motivations:
  - Detect novel attacks.
  - Provide techniques for:
    - Building the "best" possible models.
    - Predicting and characterizing the performance of the models.
- Approach:
  - Information-theoretic based measures.

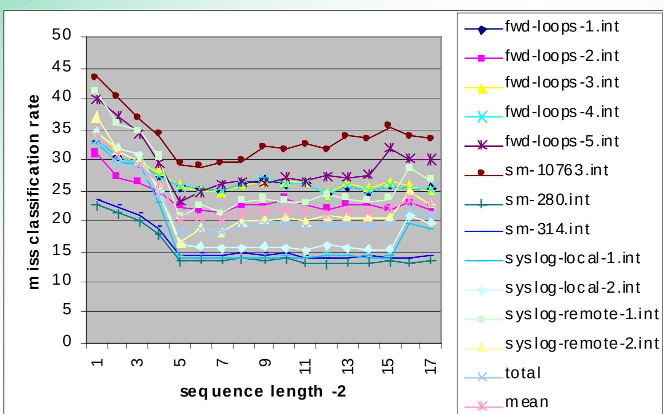
A Case Study

- Anomaly detection for Unix processes.
- UNM *sendmail* system call traces
  - "Short sequences" as normal profile
  - A classification approach:
    - Given the first *k* system calls, predict the *k*+1st system call
  - How to determine the "sequence length", *k*?

Misclassification Rate vs. Sequence Length



Misclassification Rate for Intrusion Traces



Entropy

$$H(X) = -\sum_x P(x)\log(P(x))$$
$$= -E_P[\log(P(X))]$$
$$\geq 0$$

Given random variable X, samples from X can be encoded in H(X) bits. Entropy is a measure of "uncertainty".

Conditional Entropy

$$H(Y \mid X) = -E_{P(X)}[H(Y \mid X = x)]$$
$$= -\sum_x P(x)\sum_y P(y \mid x)\log P(y \mid x)$$
$$= \sum_{x,y} P(x,y)\log P(x \mid y)$$

How "uncertain" is Y given X?

Conditional Entropy for System Call Data

• Given a system call sequence $(A_1,A_2,\ldots,A_k)$, how to predict the next system call A_{k+1}?

• Let Y be the sequence $(A_1,A_2,\ldots,A_k, A_{k+1})$, and X be the sequence $(A_1,A_2,\ldots,A_k)$,

• Conditional entropy H(Y|X):
  • how much uncertainty remains for A_{k+1} after we have seen the first *k* system calls.

How to Compute H(Y|X)

Let **S** be the set of all length ***k*+1** system call sequences in a trace,

S={x **|** x= $(A_1,A_2,\ldots,A_k, A_{k+1})$ in the trace}.

Let **|x|** be the number of occurrence of **x** in **S**.

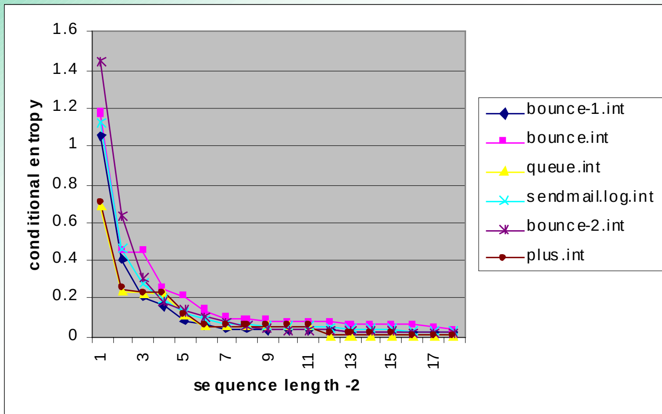
Let **y(x)** be the length-*k* subsequence of **x**.

y(x)=$(A_1,A_2,\ldots,A_k)$.

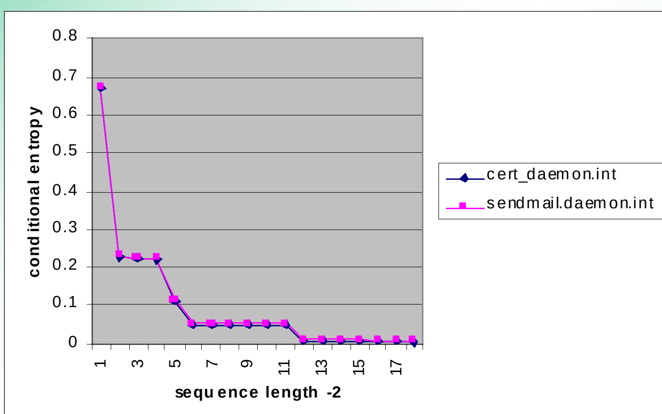
Then $$H(X \mid Y) = \sum_x \frac{|x|}{|S|} \log\left(\frac{|x|}{|y(x)|}\right)$$

because $$P(x,y) = P(x) = \frac{|x|}{|S|}$$

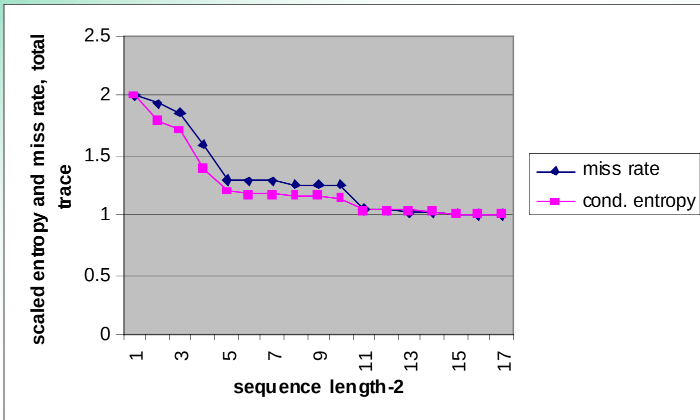
Conditional Entropy



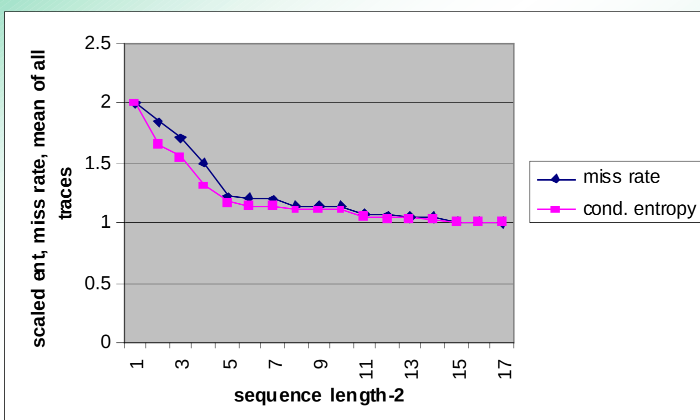
Conditional Entropy for Daemon



Condition Entropy vs. Misclassification Rate: Total Trace



Condition Entropy vs. Misclassification Rate: Mean of All Traces



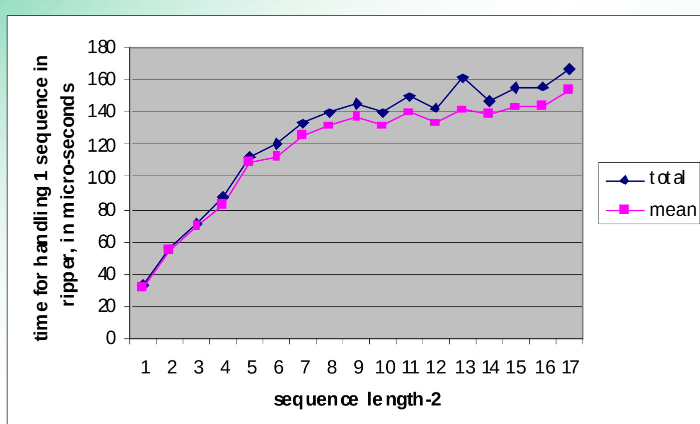
What Is the Best Sequence Length

• Get the biggest **gain** while paying the least **cost**.

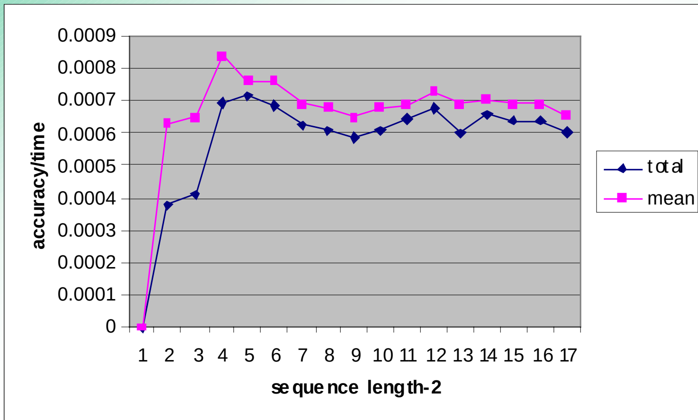
• Define the cost to be the **time to process one system call sequence**.

• Define the gain to be **accuracy of the processing**, where **accuracy=1-miss_classification_rate**.

Time to Process One Sequence



Accuracy/time

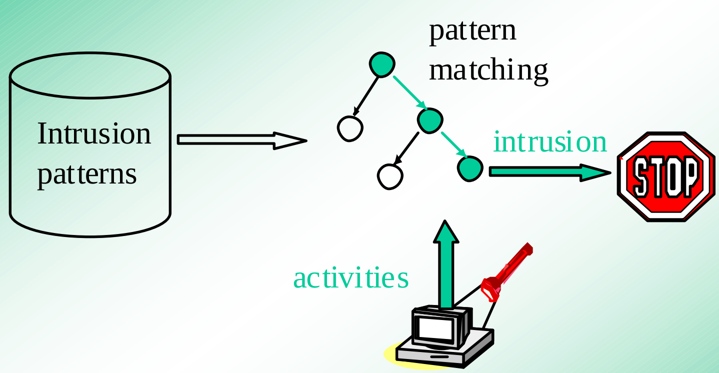


Conclusion and Future Work

- Data mining can be used to improve the ID development process.
- Future work
  - Correlation techniques for combining outputs from multiple sensors.
  - ID for emerging environments, e.g., wireless ad-hoc networks.

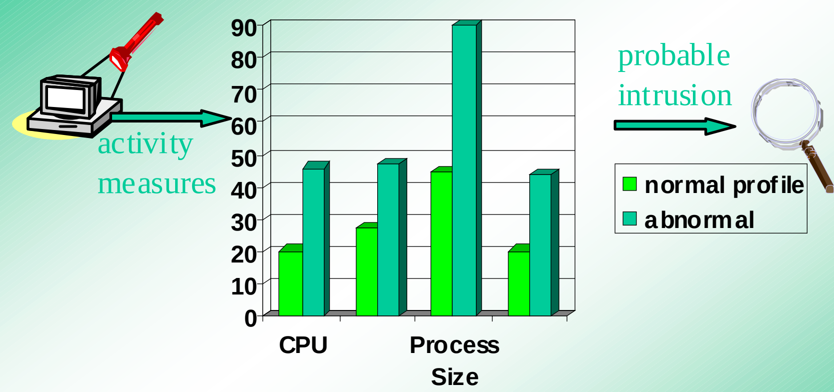
Thank You…

Misuse Detection



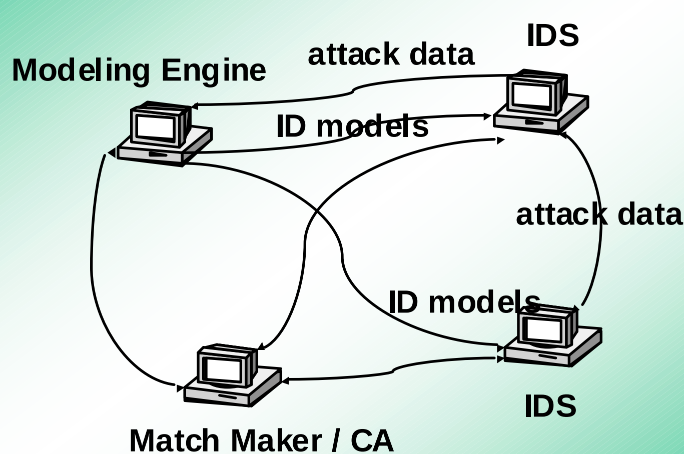
Problem: can't detect new attacks

Anomaly Detection



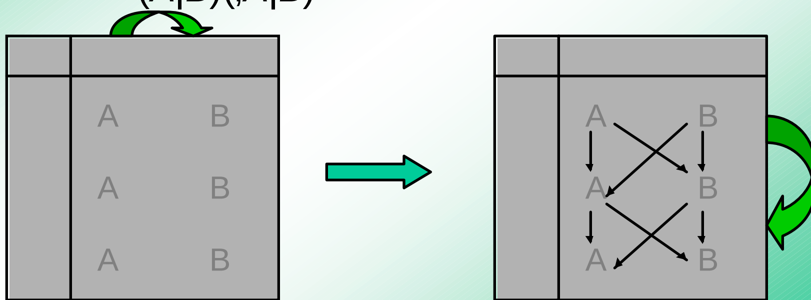
Problem: relatively high false positive rate -
anomalies can just be new normal activities.

A CIDF Architecture



Axis Attribute(s) (Continued)

– very important for frequent episodes:
  - association $A \rightarrow B$ can result in **MANY** episode rules: $(A|B)(,A|B)^* \rightarrow (A|B)(,A|B)^*$



Research in Intrusion Detection

- James Anderson (1980)
  – "malicious users", anomaly detection
- Dorothy Denning (1987)
  – generic intrusion detection system architecture
- SRI's IDES and NIDES (1992, 1993)
  – statistical and rule-based methods
- UCSB's USTAT and Purdue's IDIOT (1995)
  – state transition analysis and Colored Petri nets
- UNM's "self" models of Unix processes (1996-)
  – anomaly detection models w/ system call sequences
- MADAM ID (Lee et al. 1997-)
  – using data mining to build ID models