



PennState

CSE543 - Computer Security

Module: Intrusion Detection

Asst. Prof. Syed Rafiul Hussain

- An authorized action ...
- that exploits a vulnerability ...
- that causes a compromise ...
- and thus a successful attack.



- *Authentication and Access Control Are No Help!*

Example Intrusions

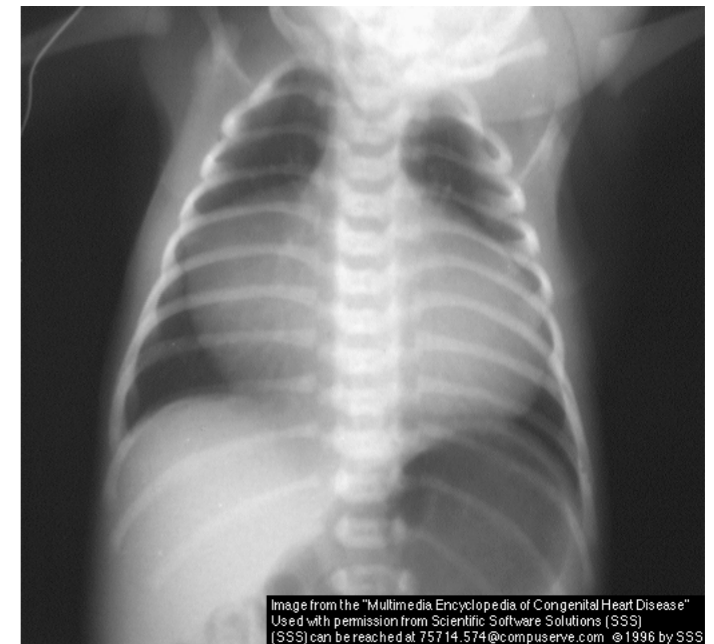
- Network

- ▶ Malformed (and unauthenticated) packet
- ▶ Let through the firewall
- ▶ Reaches the network-facing daemon
- ▶ *Can we detect intrusions from packet contents?*

- Host

- ▶ Input to daemon
- ▶ Exploits a vulnerability (buffer overflow)
- ▶ Injects attacker or reuses program code
- ▶ Performs malicious action
- ▶ *Can we detect intrusions from process behavior?*

- An IDS system finds intrusions
 - ▶ “The IDS approach to security is based on the assumption that a system will not be secure, but that violations of security policy (intrusions) can be **detected by monitoring and analyzing system behavior.**” [Forrester 98]
 - ▶ However you do it, it requires
 - Training the IDS (*training*)
 - Looking for intrusions (*detection*)
- This is active area of computer security, that has led to lots of new tools, applications, and an entire industry



- IDS's claim to detect adversary when they are in the act of attack
 - ▶ Monitor operation
 - ▶ Trigger mitigation technique on detection
 - ▶ Monitor: Network or Host (Application) **events**
- A tool that discovers intrusions “after the fact” are called ***forensic analysis*** tools
 - ▶ E.g., from system logfiles
- IDS's really refer to two kinds of detection technologies
 - ▶ ***Anomaly Detection***
 - ▶ ***Misuse Detection***



Anomaly Detection

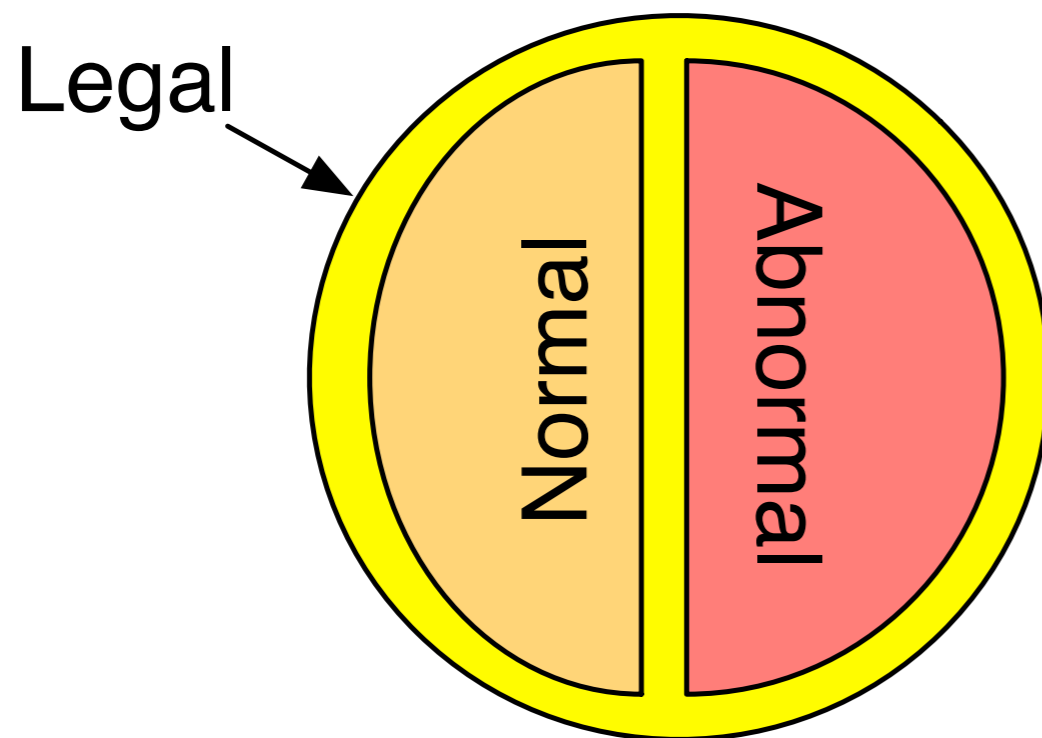
- Compares profile of normal systems operation to monitored state
 - ▶ **Hypothesis:** any attack causes enough deviation from profile (generally true?)
- **Q: How do you derive normal operation?**
 - ▶ AI: learn operational behavior from training data
 - ▶ Expert: construct profile from domain knowledge
 - ▶ Black-box analysis (vs. white or grey?)
- **Q: Is normal the same for all environments?**
- Pitfall: *false learning*



- Profile known attacks
 - ▶ Monitor operational state for known attack behaviors
 - ▶ **Hypothesis**: attacks of the same kind has enough similarity to distinguish from normal behavior
 - ▶ This is largely *pattern matching*
- **Q: Where do “known attack patterns” come from?**
 - ▶ Record: examples of known attacks
 - ▶ Expert: domain knowledge
 - ▶ AI: Learn by negative and positive feedback

The “confusion matrix”

- What constitutes a intrusion is really just a matter of definition
 - A system can exhibit all sorts of behavior

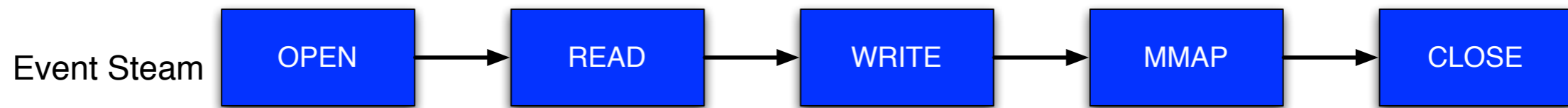


		<i>Detection Result</i>	
		T	F
<i>Reality</i>	T	True Positive	False Negative
	F	False Positive	True Negative

- Quality determined by consistency with a given definition
 - *context sensitive*

Sequences of System Calls PennState

- Forrest et al. in early-mid 90s, attempt to understand the characteristics of an intrusion



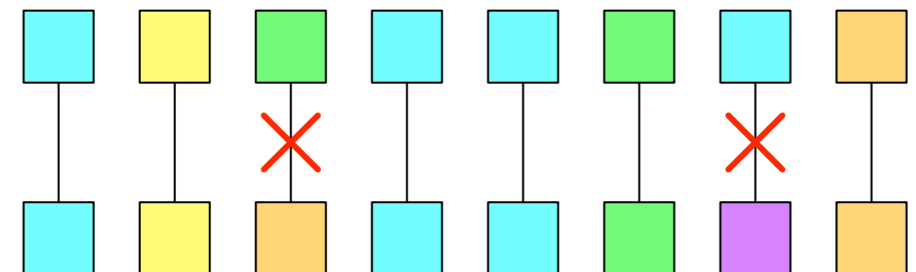
- Idea: match sequence of system calls with profiles
 - *n-grams* of system call sequences (learned)
 - ▶ Match sliding windows of sequences
 - ▶ Record the number of mismatches
 - ▶ Use n-grams of length **5, 6, 11**.
- If found, then it is normal (w.r.t. learned sequences)

- The qualitative measure of detection is the departure of the trace from the database of n-grams
- They measure how far a particular n-gram i departs by computing the minimum Hamming distance of the sample from the database (really pairwise mismatches)

$$d_{\min} = \min(d(i,j) \mid \text{for all normal } j \text{ in n-gram database})$$

this is called the *anomaly signal*.

- Result: on lpr, sendmail, etc.
 - ▶ About 1 in 100 false positive rate for lpr
 - ▶ % abnormal seqs - 1-2% for lpr attack
- Is this good?



Can You Evade Forrest?

- Can you devise a malware program that performs its malicious actions and cannot be detected by Forrest?
- How would you do that?



Can You Evade Forrest?

- Can you devise a malware program that performs its malicious actions and cannot be detected by Forrest?
- How would you do that?



- **Mimicry** - Wagner and Soto - ACM CCS 2002

"gedanken experiment"

- Assume a very good anomaly detector (99%)
- And a pretty constant attack rate, where you can observe 1 out of 10000 events are malicious



- Are you going to detect the adversary well?

Bayes' Rule

- $\Pr(x)$ function, probability of event x
 - ▶ $\Pr(\text{sunny}) = .8$ (80% of sunny day)
- $\Pr(x|y)$, probability of x given y
 - ▶ Conditional probability
 - ▶ $\Pr(\text{cavity}|\text{toothache}) = .6$
 - 60% chance of cavity given you have a toothache
 - ▶ Bayes' Rule (of conditional probability)

$$\Pr(B|A) = \frac{\Pr(A|B) \Pr(B)}{\Pr(A)}$$

- Setup

- ▶ $\Pr(T)$ is attack probability, $1/10,000$
 - $\Pr(T) = .0001$
- ▶ $\Pr(F)$ is probability of event flagging, unknown
- ▶ $\Pr(F|T)$ is 99% accurate (higher than most techniques)
 - $\Pr(F|T) = .99$, $\Pr(!F|T) = .01$, $\Pr(F|!T) = .01$, $\Pr(!F|!T) = .99$

- Deriving $\Pr(F)$

- ▶ $\Pr(F) = \Pr(F|T) * \Pr(T) + \Pr(F|!T) * \Pr(!T)$
- ▶ $\Pr(F) = (.99)(.0001) + (.01)(.9999) = .010098$

- Now, what's $\Pr(T|F)$?

The Bayesian Fallacy

- Now plug it in to Bayes Rule

$$\Pr(T|F) = \frac{\Pr(F|T) \Pr(T)}{\Pr(F)} = \frac{\Pr(.99) \Pr(.0001)}{\Pr(.010098)} = .0098$$

- So, a 99% accurate detector leads to ...
 - ▶ 1% accurate detection.
 - ▶ With 99 false positives per true positive
 - ▶ This is a central problem with IDS
- Suppression of false positives real issue
 - ▶ Open question, makes some systems unusable

Where is Anomaly Detection Useful?

System	Attack Density $P(T)$	Detector Flagging $\Pr(F)$	Detector Accuracy $\Pr(F T)$	True Positives $P(T F)$
A	0.1		0.65	
B	0.001		0.99	
C	0.1		0.99	
D	0.00001		0.99999	

$$\Pr(B|A) = \frac{\Pr(A|B) \Pr(B)}{\Pr(A)}$$

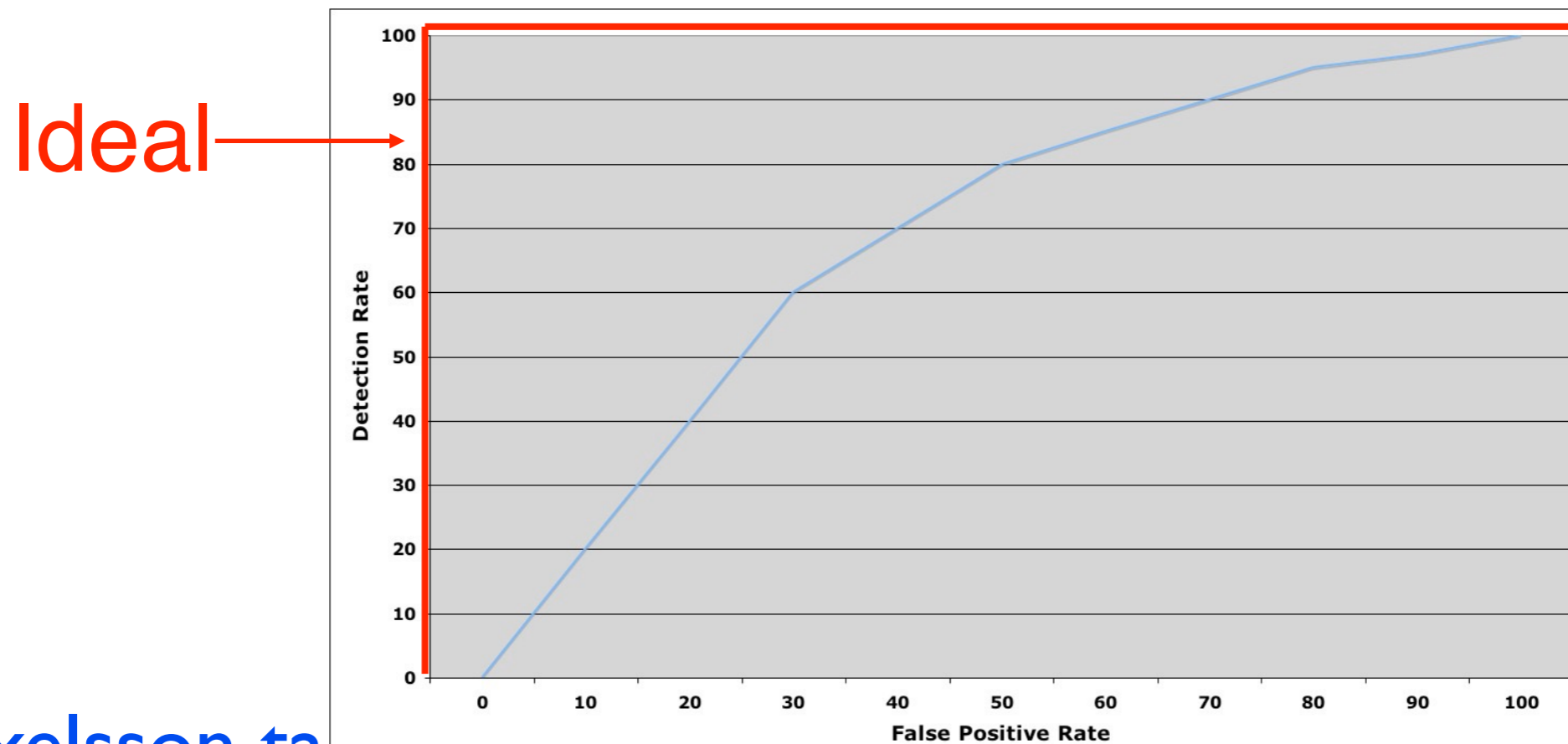
Where is Anomaly Detection Useful?

System	Attack Density $P(T)$	Detector Flagging $\Pr(F)$	Detector Accuracy $\Pr(F T)$	True Positives $P(T F)$
A	0.1	0.38	0.65	0.171
B	0.001	0.01098	0.99	0.090164
C	0.1	0.108	0.99	0.911667
D	0.00001	0.00002	0.99999	0.5

$$\Pr(B|A) = \frac{\Pr(A|B) \Pr(B)}{\Pr(A)}$$

The ROC curve

- Receiver operating characteristic
 - ▶ Curve that shows that detection/false positive ratio



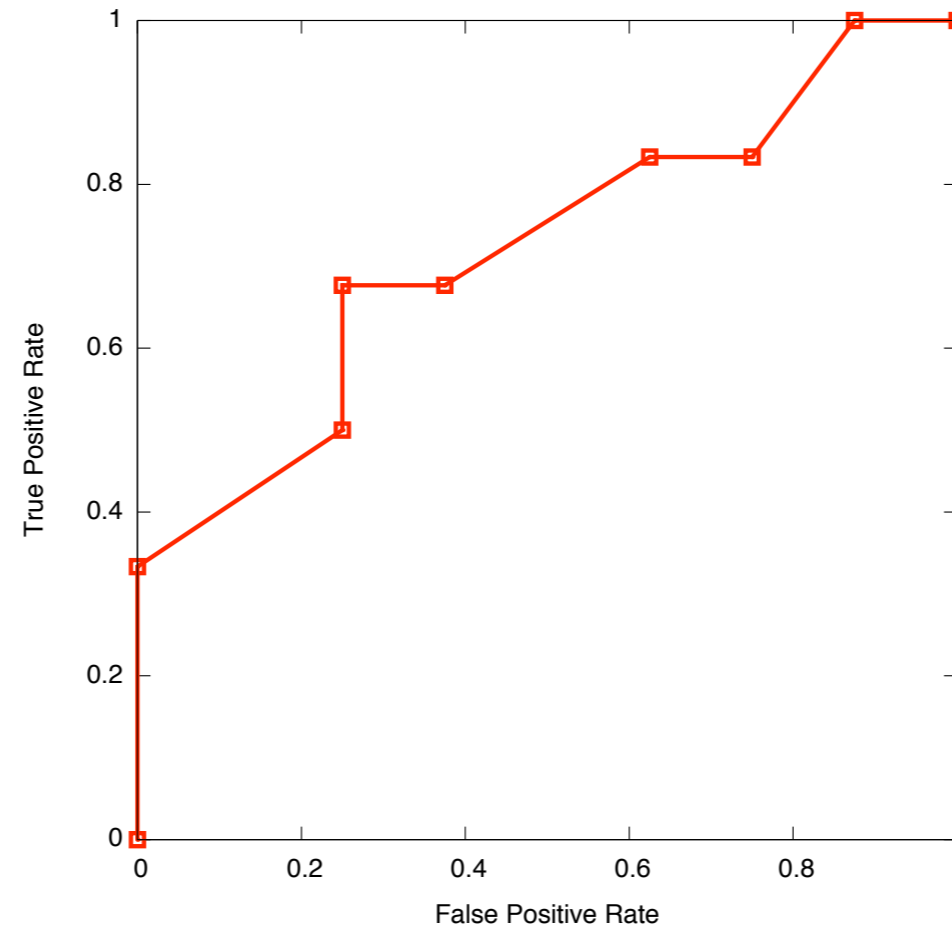
- Axelsson talks about the real problem with some authority and shows how this is not unique to CS
 - ▶ Medical example

Example ROC Curve

- You are told to design an intrusion detection algorithm that identifies vulnerabilities by solely looking at transaction length, i.e., the algorithm uses a packet length threshold T that determines when a packet is marked as an attack. More formally, the algorithm is defined:

$$D(k, T) \rightarrow [0, 1]$$

- where k is the packet length of a suspect packet in bytes, T is the length threshold, and $(0, 1)$ indicate that packet should or should not be marked as an attack, respectively. You are given the following data to use to design the algorithm.
 - ➔ attack packet lengths: 1, 1, 2, 3, 5, 8
 - ➔ non-attack packet lengths: 2, 2, 4, 6, 6, 7, 8, 9
- Draw the ROC curve.



T	0	1	2	3	4	5	6	7	8	9
TP	0	2	3	4	4	5	5	5	6	6
TP%	0.00	33.33	50.00	66.67	66.67	83.33	83.33	83.33	100.00	100.00
FP	0	0	2	2	3	3	5	6	7	8
FP%	0.00	0.00	25.00	25.00	37.50	37.50	62.50	75.00	87.50	100.00

The reality ...

- Intrusion detection systems are good at catching demonstrably bad behavior (and some subtle)
- Alarms are the problem
 - ▶ How do you suppress them?
 - ▶ and not suppress the true positives?
 - ▶ This is a limitation of *probabilistic pattern matching*, and nothing to do with bad science
- **Beware:** the fact that an IDS is not alarming does not mean the network is safe
- **All too often:** used as a tool to demonstrate all safe, but is not really appropriate for that.