CIS 4930: Secure IoT

Prof. Kaushal Kafle

Lecture 20

Derived from slides by Adwait Nadkarni, William Enck, Micah Sherr and Patrick McDaniel

Class Notes

- **2 Reminders:**

1. Homework 4 due this Tuesday.

2. Student Assessment of Instruction

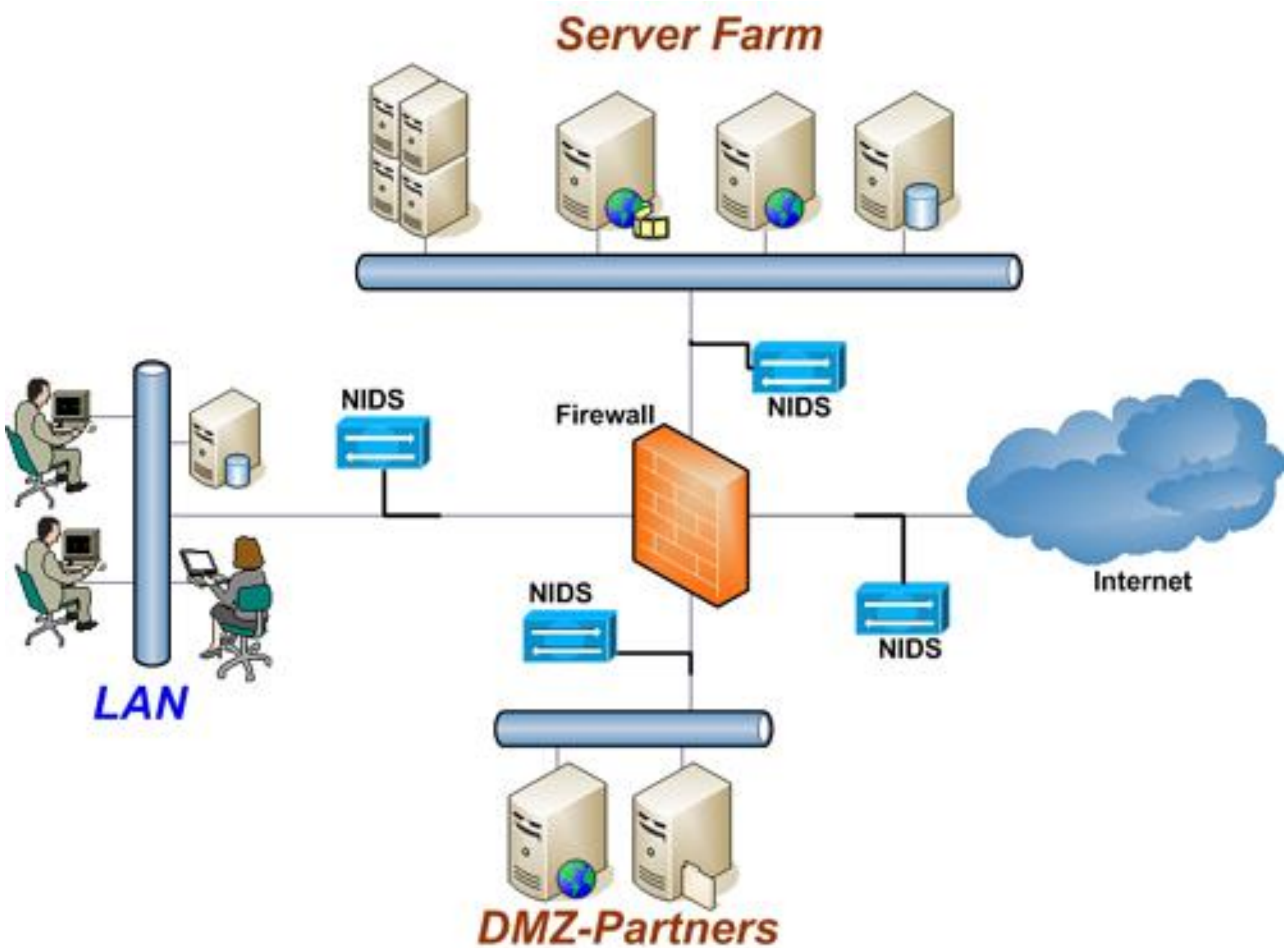
Respond to the course assessment survey.

Network Intrusion Detection Systems (NIDS)

Intrusion Detection Systems

- **Authorized eavesdropper** that listens in on network traffic
- Makes determination whether **traffic contains malware**
  - usually compares payload to virus/worm signatures
  - usually looks at only incoming traffic
- If malware is detected, IDS somehow raises an alert
- Intrusion detection is a **classification problem**

Example Setup



Detection via Signatures

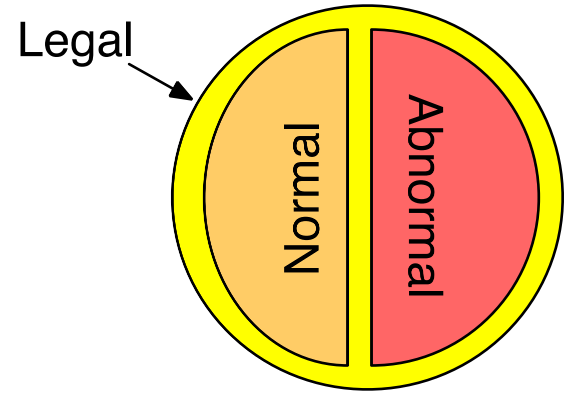
- Signature checking
  - does packet match some signature
    - suspicious headers
    - suspicious payload (e.g., shellcode)
  - great at matching known signatures
  - Low *false positive* rate: **Q: WHY?**
  - Problem: not so great for zero-day attacks -- **Q: WHY?**

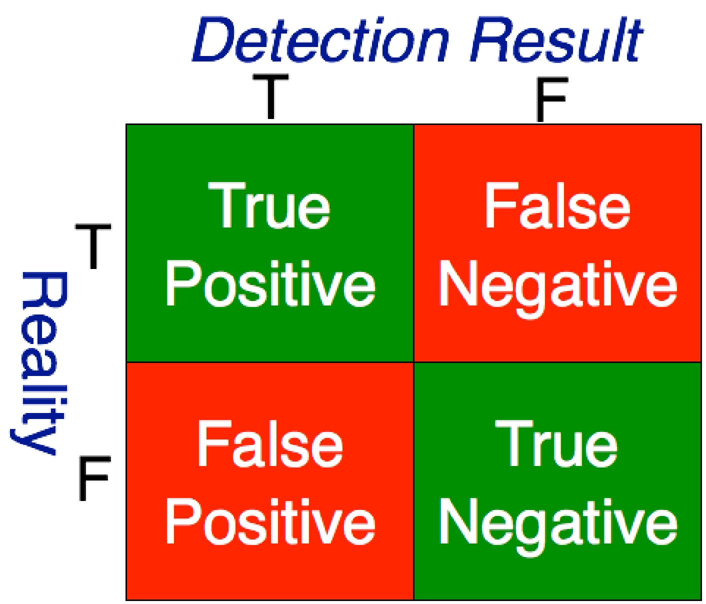
Anomaly Detection

- *Learn what "normal" looks like.*
- Frequently uses ML techniques to identify malware
- Underlying assumption: malware will look different from non-malware
- **Supervised learning**
  - IDS requires learning phase in which operator provides pre-classified ***training data*** to learn patterns
  - {good, 80, "GET", "/", "Firefox"}
  - {bad, 80, "POST", "/php-shell.php?cmd='rm -rf /'", "Evil Browser"}
  - ML technique builds model for classifying never-before-seen packets
  - Problem: *False Learning*
  - Problem: is new malware going to look like training malware?

Confusion Matrix

- What constitutes an intrusion/anomaly is really just a matter of definition
  - A system can exhibit all sorts of behavior
- Quality determined by the consistency with a given definition
  - Context-sensitive (i.e., what is "positive/true"?)
- *These concepts are quite relevant to your app analysis project results!*





Metrics

- **True positives** (TP): number of correct classifications of malware
- **True negatives** (TN): number of correct classifications of non-malware
- **False positives** (FP): number of incorrect classifications of non-malware as malware
- **False negatives** (FN): number of incorrect classifications of malware as non-malware

Metrics

(from perspective of detector)

Reality \ Detection Result	T	F
T	True Positive	False Negative
F	False Positive	True Negative

- **False positive rate**:

$$FPR = \frac{FP}{FP + TN} = \frac{\#\text{ benign marked as malicious}}{\text{total benign}}$$

- **True negative rate**:

$$TNR = 1 - FPR = \frac{TN}{FP + TN} = \frac{\#\text{ benign unmarked}}{\text{total benign}}$$

- **False negative rate:**

$$FNR = \frac{FN}{FN + TP} = \frac{\#\text{ malicious not marked}}{\text{total malicious}}$$

- **True positive rate:**

$$TPR = 1 - FNR = \frac{TP}{FN + TP} = \frac{\#\text{ malicious correctly marked}}{\text{total malicious}}$$

Base Rate Fallacy

- Occurs when we assess P(X|Y) without considering prior probability of X and the total probability of Y
- **Example**:
  - ***Base rate*** of malware is 1 packet in a 10,000
  - Intrusion detection system is 99% accurate (given known samples). Let's assume this means:
    - 1% false positive rate (benign marked as malicious 1% of the time)
    - 1% false negative rate (malicious marked as benign 1% of the time)
  - Packet X is marked by the NIDS as malware. *What is the probability that packet X actually is malware?*
    - Let's call this the "**true alarm rate**," because it is the rate at which the raised alarm is actually true.

Base-rate Fallacy in the real world

Health Nerd
@GidMK

Follow

So, according to this, the false positive rate for the Apple Watch in detecting atrial fibrillation is 0.04% (99.6% correct)

This means that, on average, Apple Watches will be wrong more than 80% of the time

Sound counterintuitive? This is the issue with population screening

> **STAT** @statnews
> Apple submitted two studies to FDA to get clearance for the new Apple Watch EKG app. Here's the data. buff.ly/2QuhGmG

5:28 AM - 14 Sep 2018

Bayes' Rule

- Pr(*x*) function -> probability of event *x*
  - Pr(sunny) = .8 (80% of days are sunny)
- Pr(x|y), probability of x given y
  - Conditional probability
  - Bayes' Rule
    (of conditional probability)

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

- ***Example:***
- Pr(cavity|toothache) = 0.6
  - 60% chance of cavity given you have a toothache
- Assume: Pr(cavity) = 0.5,
  Pr(toothache) = 0.1
- What is Pr(toothache|cavity)?

$$\frac{0.6 \times 0.1}{0.5} = 0.12$$

Base Rate Fallacy

- How do we find the true alarm rate? [i.e., Pr(IsMalware|MarkedAsMalware)]

$$Pr(IsMalware|MarkedAsMalware) = \frac{Pr(MarkedAsMalware|IsMalware) \cdot Pr(IsMalware)}{Pr(MarkedAsMalware)}$$

- We know:
  - 1% false positive rate (benign marked as malicious 1% of the time); TNR= 99%
  - 1% false negative rate (malicious marked as benign 1% of the time); TPR= 99%
  - *Base rate* of malware is 1 packet in 10,000
- What is?
  - Pr(MarkedAsMalware|IsMalware) = TPR = 0.99
  - Pr(IsMalware) = Base rate = 0.0001
  - Pr(MarkedAsMalware) = ?

$$Pr(MarkedAsMalware|IsMalware) = \frac{\#\text{ malicious correctly marked}}{\text{total malicious}} = \frac{TP}{FN + TP} = TPR$$

total probability that a packet is flagged as malware *(either because it is actually malware or because it is benign but flagged as malware)*

Base Rate Fallacy

- How do we find Pr(MarkedAsMalware)? -> sum of two mutually exclusive cases

= Pr(MarkedAsMalware|IsMalware) * Pr(IsMalware) + Pr(MarkedAsMalware|IsNotMalware) * Pr(IsNotMalware)

- So what is..
  - Pr(IsMalware) = base rate = 0.0001
  - Pr(IsNotMalware) = ? 1 – Pr(IsMalware) = 0.999
  - Pr(MarkedAsMalware|IsMalware) = TPR = 0.99
  - Pr(MarkedAsMalware|IsNotMalware) = ? FPR = 0.01

$$Pr(MarkedAsMalware|IsNotMalware) = \frac{\text{\# benign marked as malicious}}{\text{total benign}}$$

FP

total probability that a packet is flagged as malware *(either because it is actually malware or because it is benign but flagged as malware)*

- S .01 * 0.9999 ~= 0.01

Base Rate Fallacy

- How do we find the true alarm rate? [i.e., Pr(IsMalware|MarkedAsMalware)]

$$Pr(IsMalware|MarkedAsMalware) = \frac{Pr(MarkedAsMalware|IsMalware) \cdot Pr(IsMalware)}{Pr(MarkedAsMalware)}$$

$$= \frac{0.99 \cdot 0.0001}{0.01} = 0.0099$$

- Therefore *only about 1% of alarms are actually malware!*
  - ***What does this mean for network administrators?***

Base Rate Fallacy

(summary)

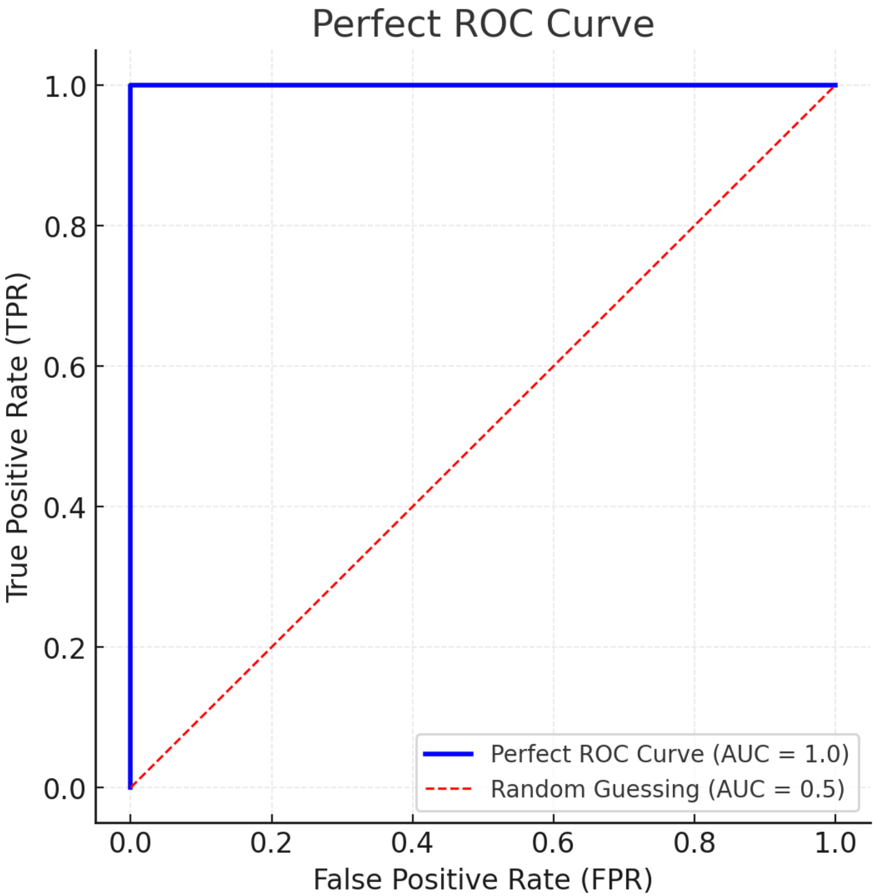
- Let Pr(M) be the probability that a packet is actually malware (the base rate)
- Let Pr(A) be the probability that that the IDS raises an alarm (unknown)
- Assume we also know for the IDS
  - Pr(A|M) = TPR = 1 - FNR
  - Pr(A|!M) = FPR
- Then the true alarm rate is

$$Pr(M|A) = \frac{Pr(A|M) \cdot Pr(M)}{Pr(A|M) \cdot Pr(M) + Pr(A|!M) \cdot Pr(!M)}$$

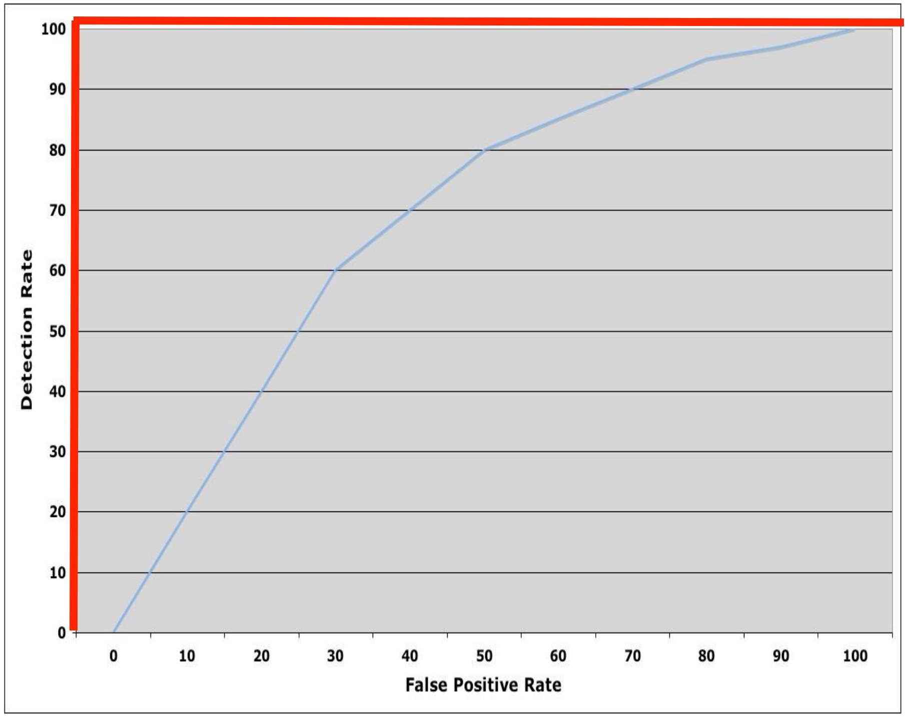
- **Note the strong influence of Pr(M)**

The ROC curve

- Receiver Operating Characteristic (ROC)
  - Curve that shows that detection/false positive ratio (for a binary classifier system as its discrimination threshold is varied)



Reality



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 (i.e., less than or equal to length T). 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

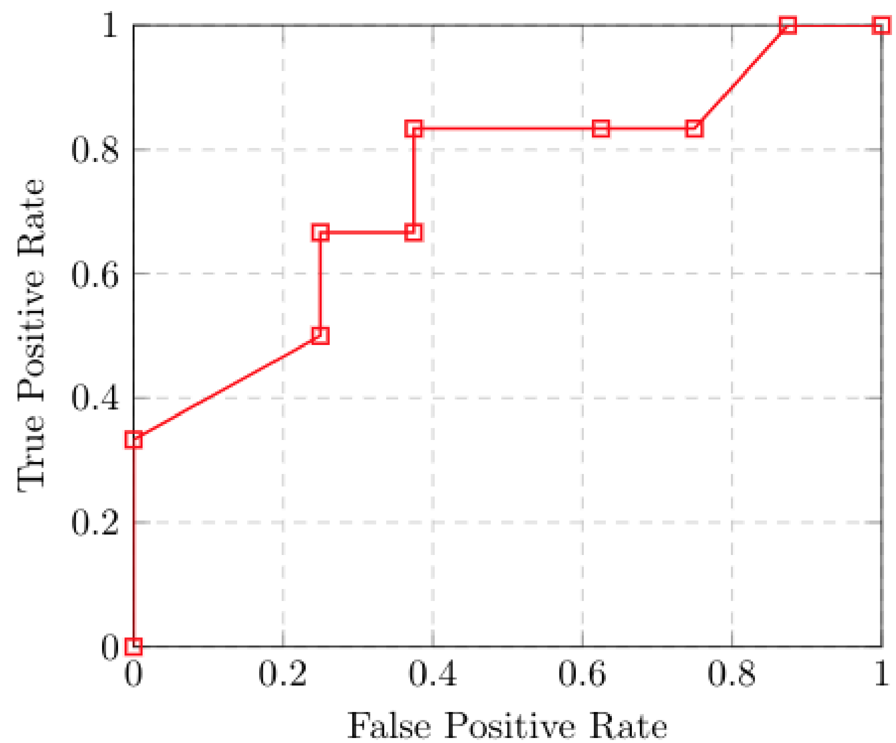
- Draw the ROC curve.

Solution

- attack packet lengths: 1, 1, 2, 3, 5, 8
- non-attack packet lengths: 2, 2, 4, 6, 6, 7, 8, 9

$$TP\% = TPR = \frac{TP}{TP + FN}$$

$$FP\% = FPR = \frac{FP}{FP + TN}$$



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

Problems with IDSes

- VERY difficult to get both good recall and precision
- Malware comes in small packages
- Looking for one packet in a million (billion? trillion?)
- If insufficiently sensitive, IDS will miss this packet (low recall)
- If overly sensitive, too many alerts will be raised (low precision)

Defenses thus far

- **Firewalls** and **Intrusion Prevention Systems** prevent malicious packets from entering the network (in theory)
- **Intrusion Detection Systems** alert network administrators to intrusion attempts
- Both systems work best when malware is well-understood and easily fingerprinted

How do we learn about and study malware?

Honeypots

- **Honeypot:** a controlled environment constructed to trick malware into thinking it is running in an unprotected system
  - collection of decoy services (fake mail, web, ftp, etc.)
  - decoys often mimic behavior of unpatched and vulnerable services

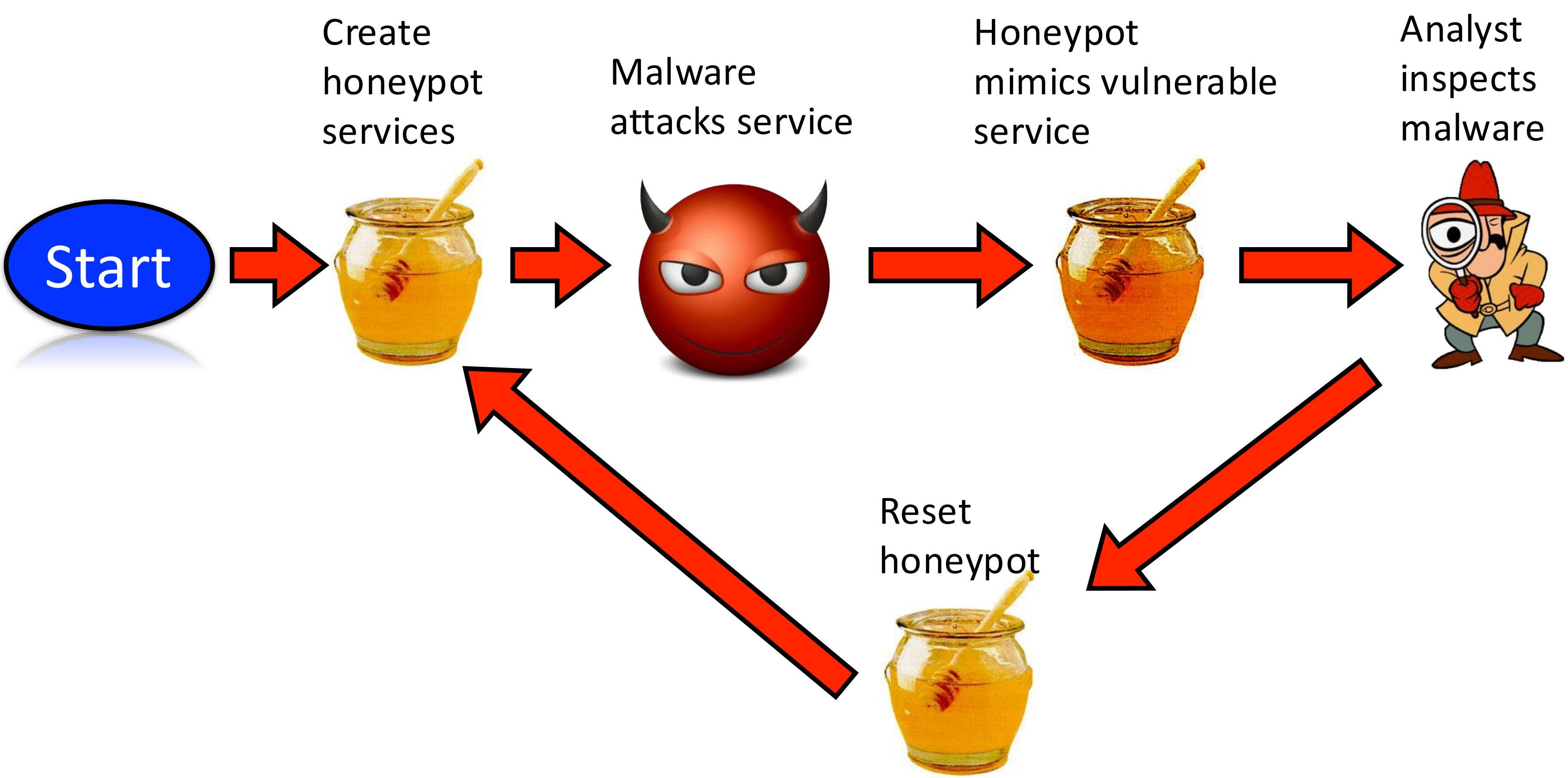
Honeypots

- Three main uses:
  - forensic analysis: better understand how malware works; collect evidence for future legal proceedings
  - risk mitigation:
    - provide “low-hanging fruit” to distract attacker while safeguarding the actually important services
    - tarpits: provide very slow service to slow down the attacker
  - malware detection: examine behavior of incoming request in order to classify it as benign or malicious

Honeypots

- Two main types:
  - **Low-interaction:** emulated services
    - inexpensive
    - may be easier to detect
  - **High-interaction:** no emulation; honeypot maintained inside of real OS
    - expensive
    - good realism

Example Honeypot Workflow



Examining Malware

- **Trace system calls:**
  - most OSes support method to trace sequence of system calls
    - e.g., ptrace, strace, etc.
  - all “interesting” behavior (e.g., networking, file I/O, etc.) must go through system calls
  - capturing sequence of system calls (plus their arguments) reveals useful info about malware’s behavior

Tracing System Calls

```
@ubuntu:~$ strace ls
execve("/usr/bin/ls", ["ls"], 0x7ffcece74500 /* 51 vars */) = 0
brk(NULL)                               = 0x5599795d0000
access("/etc/ld.so.preload", R_OK)      = -1 ENOENT (No such file or directory)
openat(AT_FDCWD, "/etc/ld.so.cache", O_RDONLY|O_CLOEXEC) = 3
fstat(3, {st_mode=S_IFREG|0644, st_size=70930, ...}) = 0
mmap(NULL, 70930, PROT_READ, MAP_PRIVATE, 3, 0) = 0x7efc009a9000
close(3)                                = 0
openat(AT_FDCWD, "/lib/x86_64-linux-gnu/libselinux.so.1", O_RDONLY|O_CLOEXEC) = 3
read(3, "\177ELF\2\1\1\0\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0 h\0\0\0\0\0\0"..., 832) = 832
fstat(3, {st_mode=S_IFREG|0644, st_size=158928, ...}) = 0
mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x7efc009a7000
mmap(NULL, 170192, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x7efc0097d000
mprotect(0x7efc00983000, 131072, PROT_NONE) = 0
mmap(0x7efc00983000, 98304, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x6000) = 0x7efc00983000
mmap(0x7efc0099b000, 28672, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1e000) = 0x7efc0099b000
mmap(0x7efc009a3000, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x25000) = 0x7efc009a3000
mmap(0x7efc009a5000, 6352, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0) = 0x7efc009a5000
close(3)                                = 0
openat(AT_FDCWD, "/lib/x86_64-linux-gnu/libc.so.6", O_RDONLY|O_CLOEXEC) = 3
read(3, "\177ELF\2\1\1\3\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0\200l\2\0\0\0\0\0"..., 832) = 832
fstat(3, {st_mode=S_IFREG|0755, st_size=2000480, ...}) = 0
```

Examining Malware

- **Observe filesystem changes and network IO:**
  - “diff” the filesystem before and after
    - which files are the malware reading/writing?
  - capture network packets
    - to whom is the malware communicating

Examining Malware

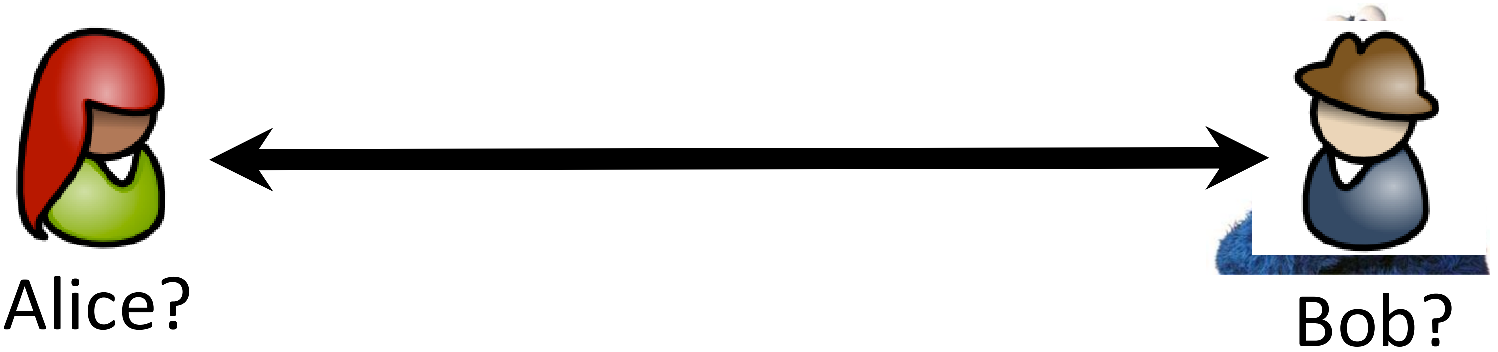
- **Utilize hidden kernel module:**
  - To capture all activity
  - Challenge: malware may already have root access!
  - Challenge: may be encrypted so can't look at binary

Challenges

- Honeypot *must resemble actual machine*
  - simulate actual services (Apache, MySQL, etc.)
  - but not too much... bad form to actually help propagate the worm (legal risks!)
- Some worms do a reasonably good job of detecting honeypots

User Authentication


Username : admin
Password : admin



Authentication

What is Authentication?

- Short answer: establishes identity
  - Answers the question: To whom am I speaking?

- Long answer: evaluates the authenticity of identity proving credentials.
  2 parts:
  - Credential – is proof of identity
  - Evaluation – process that assesses the correctness of the association between credential and claimed identity
    - for some purpose
    - under some policy (what constitutes a good cred.?)

Examples of Authentication

- Two broad types of authentication
  - User authentication
    - Allow a user to prove his/her identity to another entity (e.g., a system, a device).
  - Message authentication
    - Verify that a message has not been altered without proper authorization.

Authentication Mechanisms

- Password-based authentication
  - Use a secret quantity (the password) that the prover states to prove he/she knows it.
  - Threat: password guessing/dictionary attack

Authentication Mechanisms (Cont'd)

- Address-based authentication
  - Assume the identity of the source can be inferred based on the network address from which packets arrive (aka *authentication by assertion*)
  - Adopted early in UNIX and VMS
- Berkeley *rtools* (*rsh*, *rlogin*, etc)
  - */etc/hosts.equiv* file
    - List of computers
  - Per user *.rhosts* file
    - List of <computer, account>
- Threat
  - Spoof of network address
    - Not authentication of source addresses

Authentication Mechanisms (Cont’d)

- Cryptographic authentication protocols
  - Basic idea:
    - A prover proves some information by performing a cryptographic operation on a quantity that the verifier supplies.
  - Usually reduced to the knowledge of a secret value
    - A symmetric key
    - The private key of a public/private key pair

Why authentication?

- We live in a world of rights, permissions, and duties
  - Authentication establishes our identity so that we can obtain the set of rights
  - E.g., we establish our identity with a store by providing a valid credit card which gives us rights to purchase goods
    - this is a *physical* authentication system
    - ***Threats?***

Why authentication?

- Same in online world, just with different constraints
  - Vendor/customer are not physically co-located, so we must find other ways of providing identity
    - e.g., by providing credit card number ~ electronic authentication system
  - Risks (for customer and vendor) are different
    - Q: How so?
  - **Computer security is critically dependent on the proper design, management, and application of authentication systems**