University of Colorado **Boulder**

CSCI 3403 INTRO TO CYBERSECURITY

Lecture: 5-2

Topic: Authentication

Presenter: Matt Niemiec

Announcements

- Project 2 is going up soon
- Concerns with unprofessional behavior on Piazza
  - “Love prospers when a fault is forgiven, but dwelling on it separates close friends”
  - I’m more sympathetic when you communicate nicely!

Authentication

What is Authentication?

- Prove a user is who they say they are
- Different than authorization
- Verify your identity
  - Whatever that is!

Four Means of Authenticating a User

- Something you have
  - Credit card, badge, smart card, key, etc.
- Something you know
  - PIN, password, security questions, etc.
- Something you are
  - Fingerprint, retinal scan, facial recognition, etc.
- Something you do
  - Voice pattern, typing rhythm, gait analysis, etc.

Something You Know

Passwords

- Easy to verify
- Choose between
  - Easy to remember
  - Difficult to guess
- Should be hashed in case of a breach

Rainbow tables

- Pre-compute common hashes (>30GB worth!)
- Useful for offline or online?

Salting

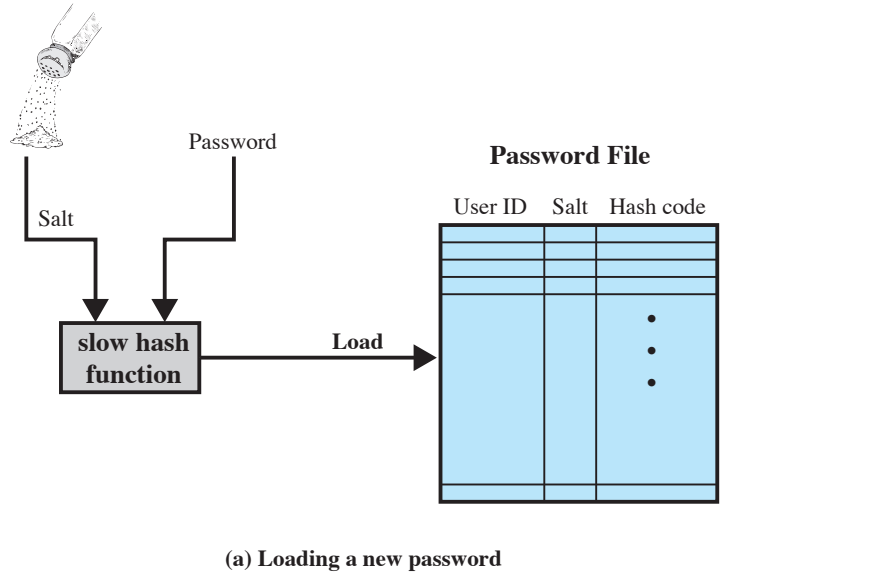


Figure 3.3 UNIX Password Scheme

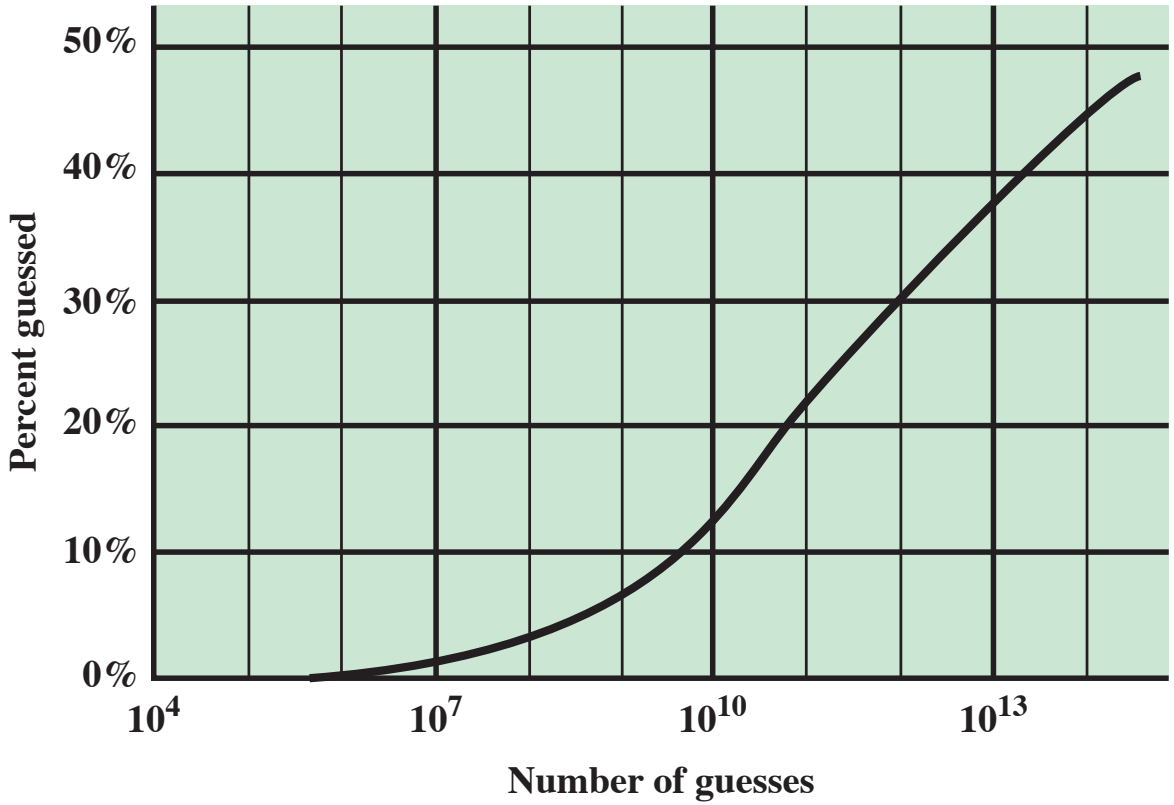


Figure 3.4 The Percentage of Passwords Guessed After a Given Number of Guesses


John
the
Ripper

Preventing Common Passwords

- Say we want to prevent *n* passwords
- Idea: Every time a user submits a password, we scan through *n* known bad passwords to to check if it's bad
- Current solution: O(?) runtime and O(?) space

Preventing Common Passwords

- Say we want to prevent *n* passwords
- Idea: Every time a user submits a password, we scan through *n* known bad passwords to to check if it's bad
- Current solution: O(*n*) runtime and O(*n*) space

Preventing Common Passwords

- Say we want to prevent *n* passwords
- Idea: Every time a user submits a password, we scan through *n* known bad passwords to to check if it's bad
- Current solution: O(*n*) runtime and O(*n*) space
- Improve: Can we get down to O(1) runtime?

Preventing Common Passwords

- Say we want to prevent *n* passwords
- Idea: Every time a user submits a password, we scan through *n* known bad passwords to to check if it's bad
- Current solution: O(*n*) runtime and O(*n*) space
- Improve: Can we get down to O(1) runtime?
- Improve again: Can we get down to O(1) space?*

*Still would scale in order to be effective, but saves a lot of space regardless

Solution: Bloom Filters

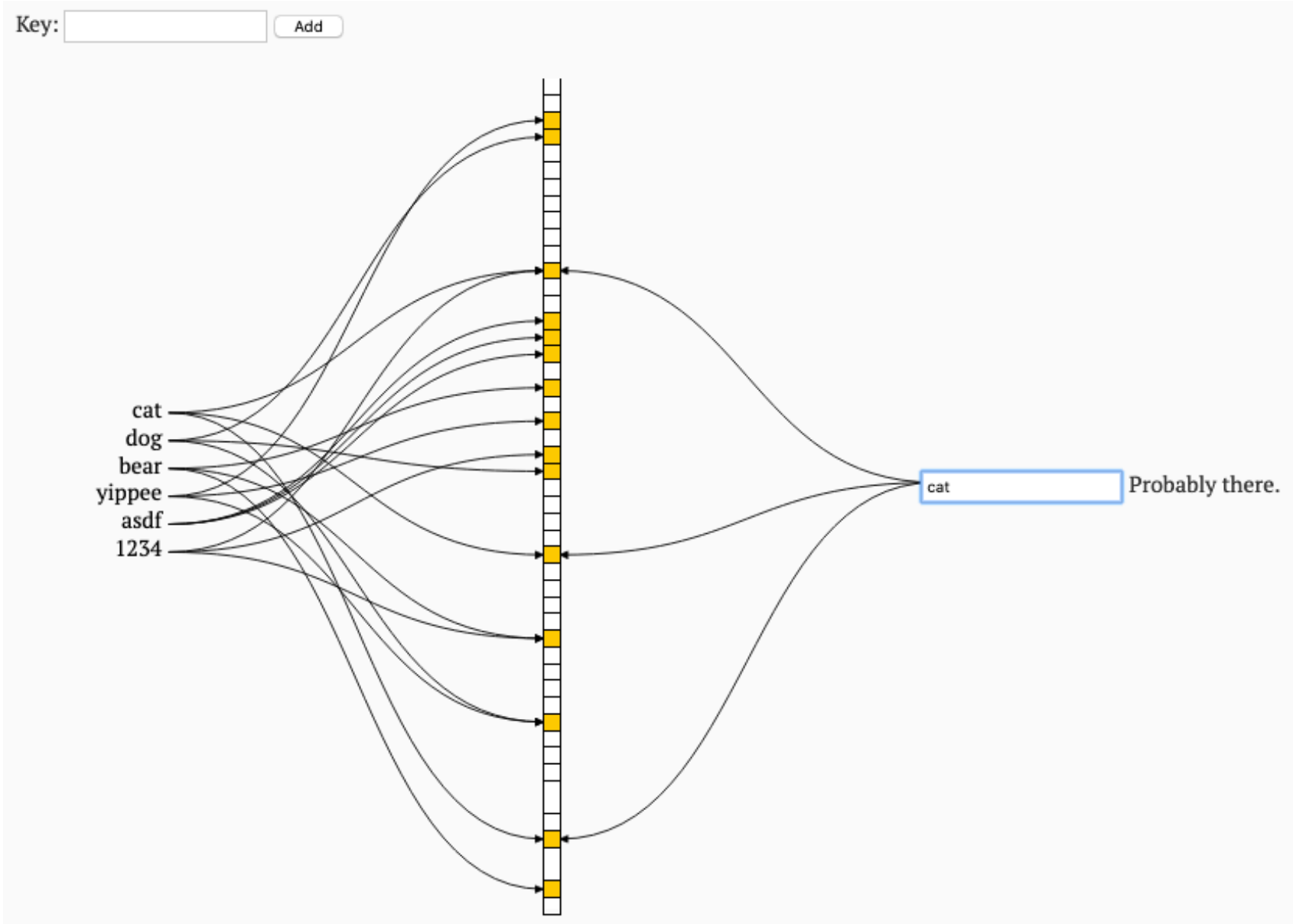
Creation

- Use 3* hash functions to hash password
- Mark each hash location as “set”
- Repeat for all known bad passwords

Checking

- Take user’s entered password
- Hash with same 3 hash functions
- If all 3 locations are marked, don’t permit the password

* Actual number can be adjusted


Key:
Add
cat
dog
bear
yippee
asdf
1234
cat
Probably there.

Something You Have

Something You Have

- Advantages
  - Very practical
  - Not easily misunderstood
  - Requires physical interaction to steal
- Disadvantages
  - Can be stolen
  - Can be lost
  - May be replicated

Something You Are/Do

Something You Are

- “Static biometrics”
- Retinal scan, fingerprint
- Advantages
  - Requires a physical presence
- Disadvantages
  - Difficult to change

Something You Do

- “Dynamic biometrics”
- Gait, typing rhythm, voice, etc.

Something You Are/Do

- Requires physical presence
- Typically very accurate
  - How accurate is accurate enough?

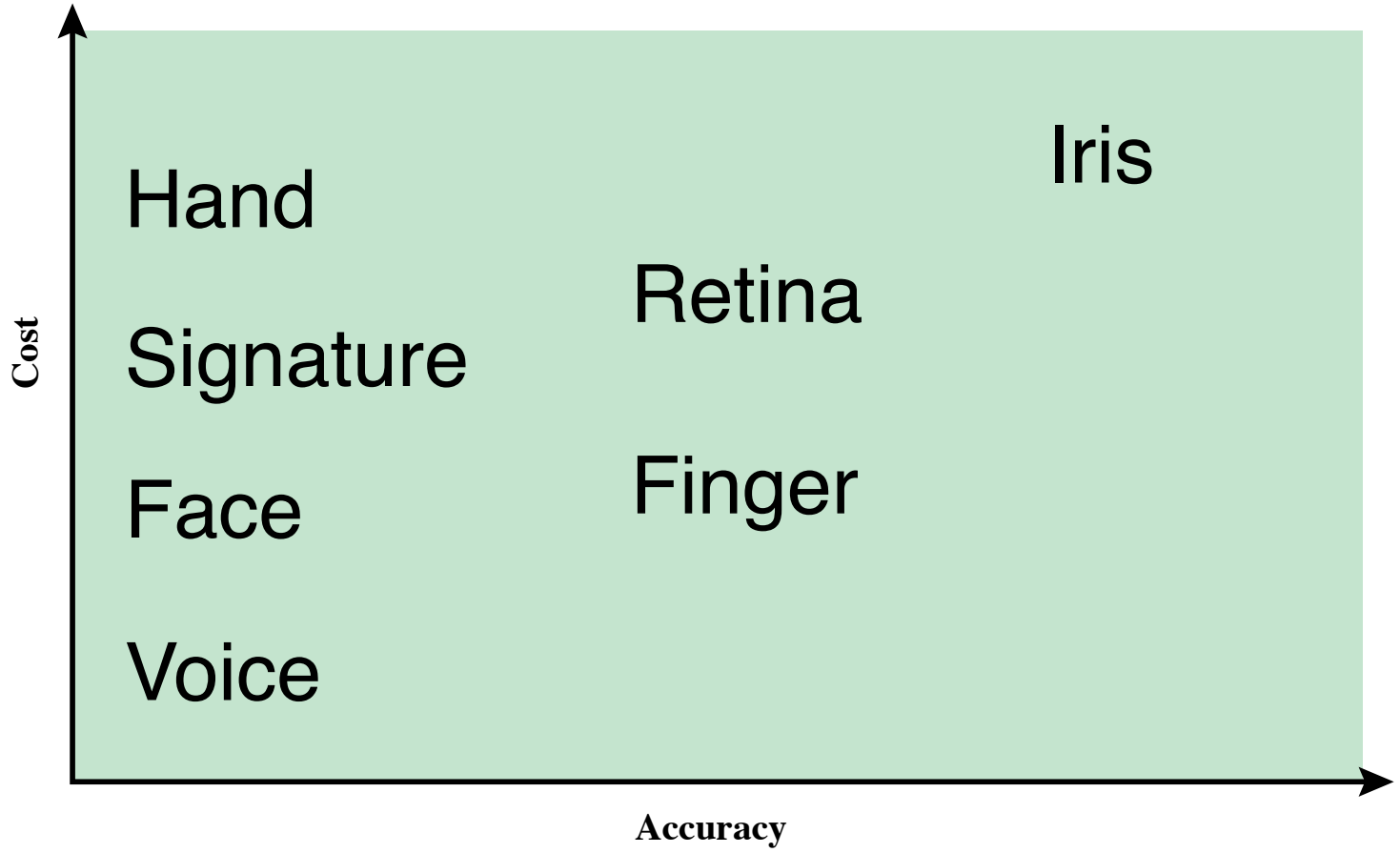


Figure 3.8 Cost Versus Accuracy of Various Biometric Characteristics in User Authentication Schemes.

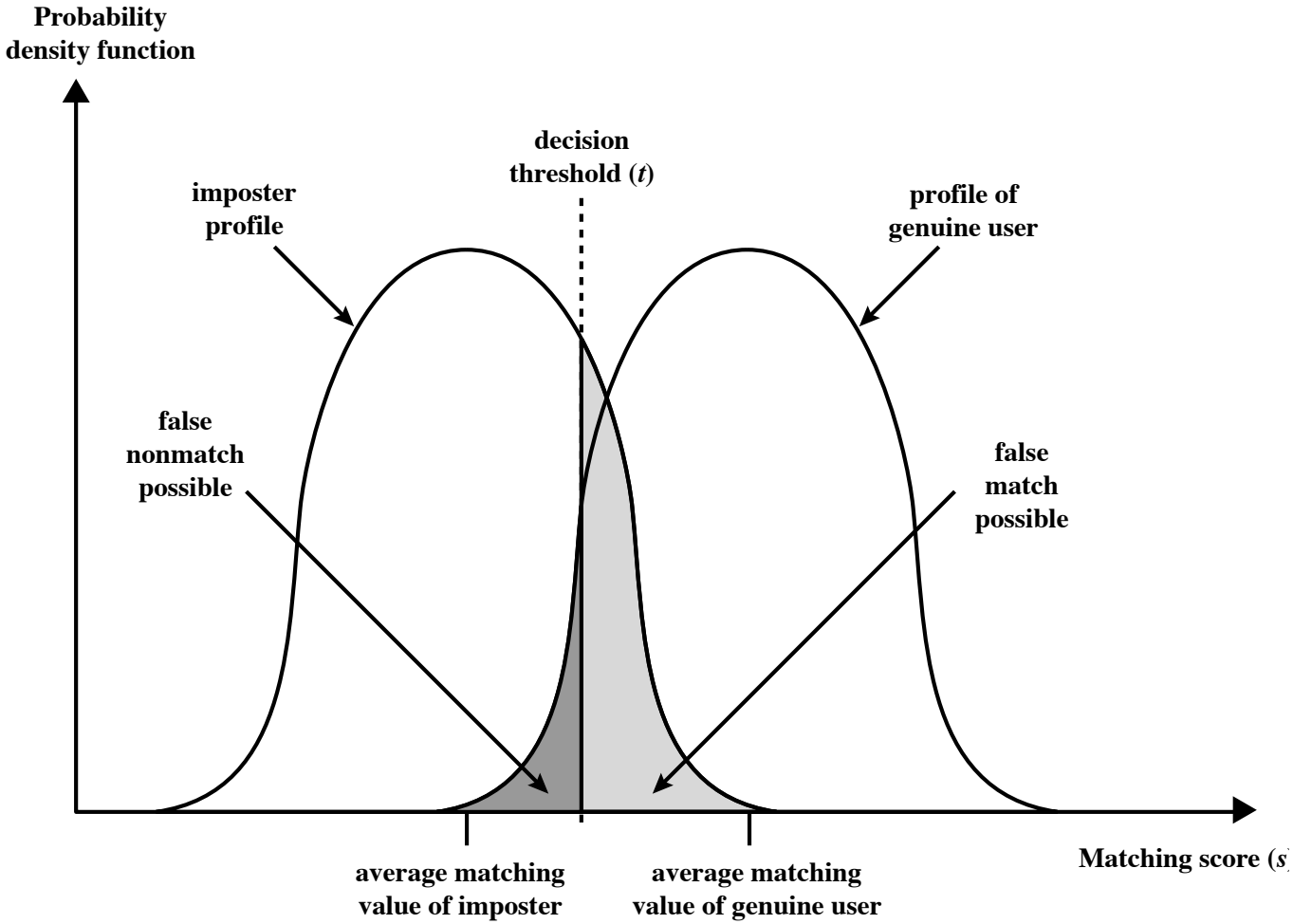
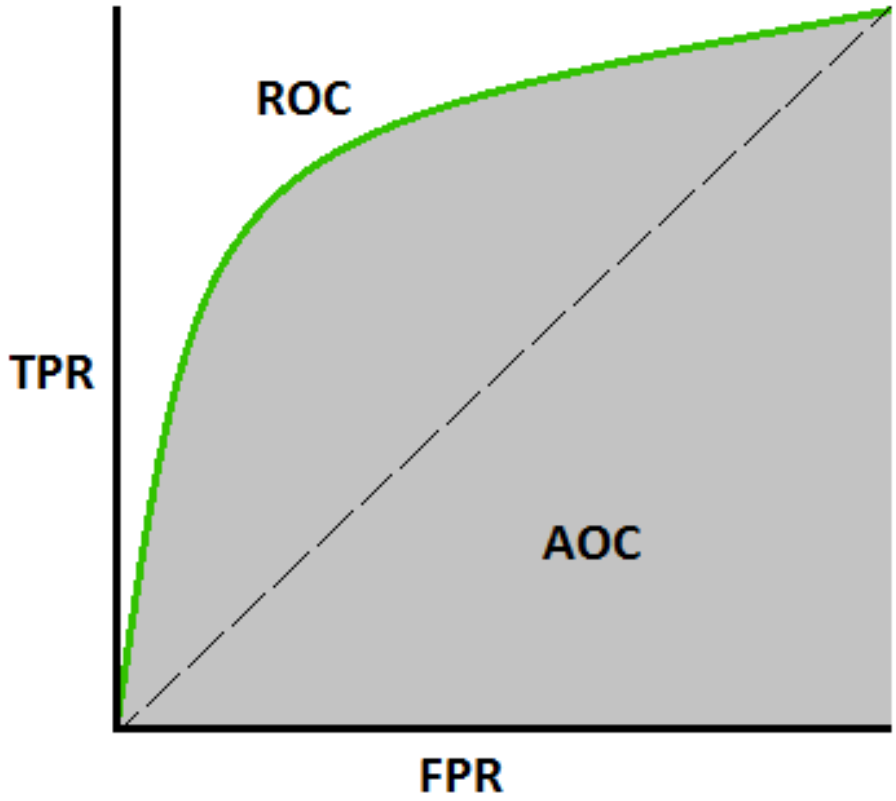
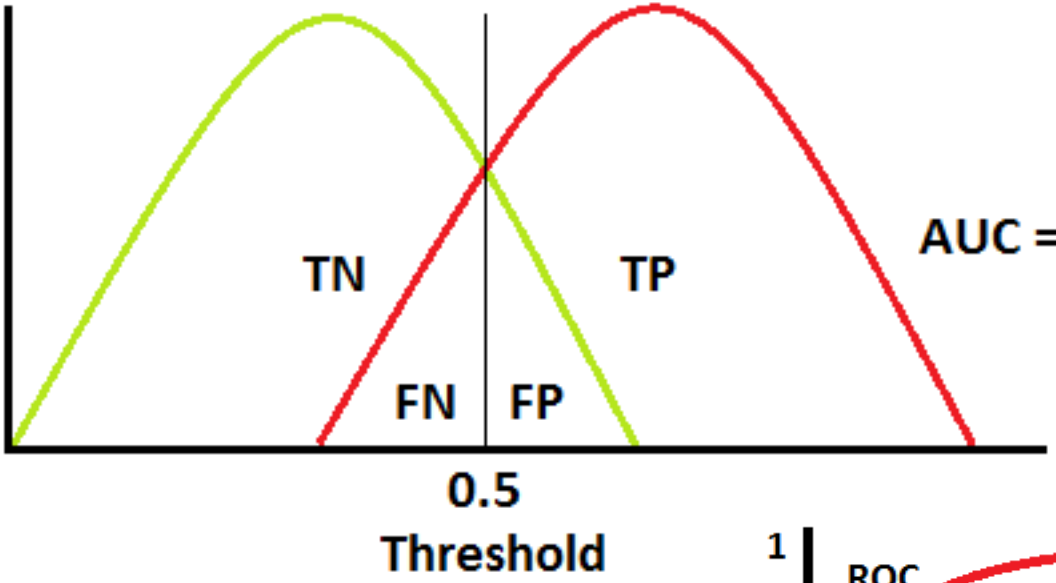


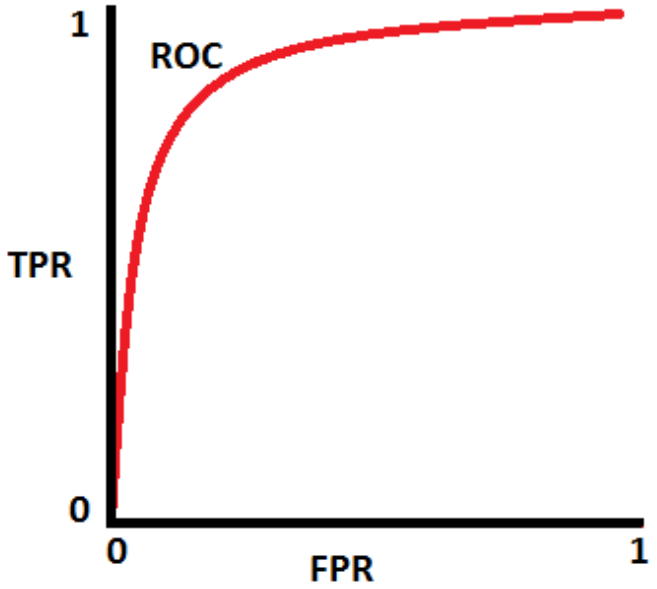
Figure 3.10 Profiles of a Biometric Characteristic of an Imposter and an Authorized Users In this depiction, the comparison between presented feature and a reference feature is reduced to a single numeric value. If the input value (s) is greater than a preassigned threshold (t), a match is declared.

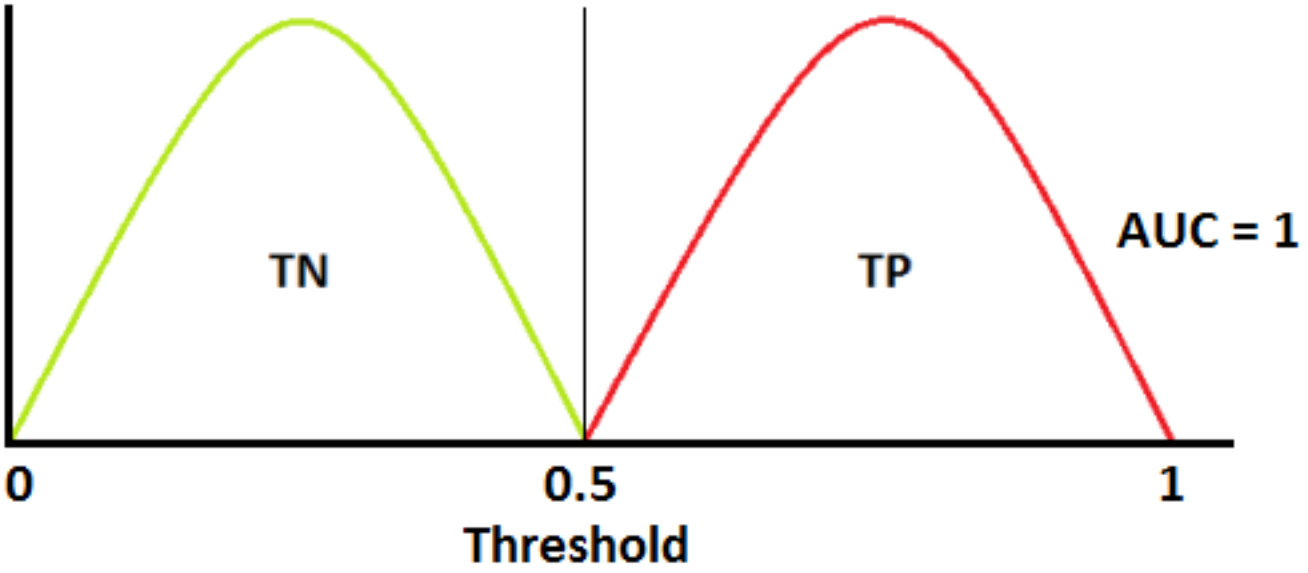
ROC Curves

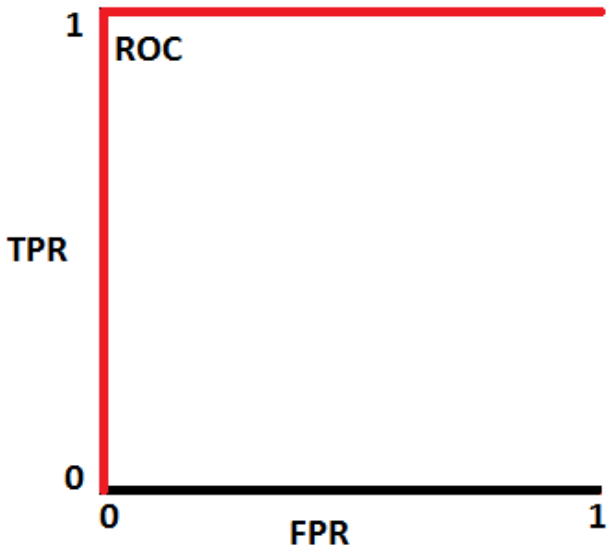
- Way of visually expressing the previous slide
- Gives simple way to analyze where to put the threshold

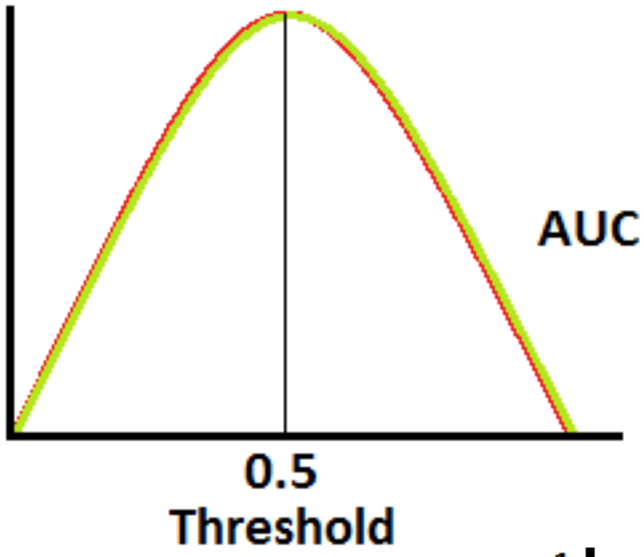


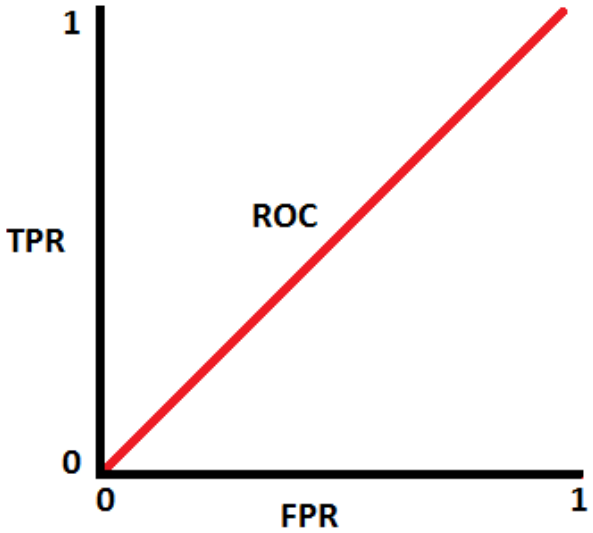

TN
TP
FN
FP
AUC = 0.7
0.5
Threshold


1
ROC
TPR
0
0
1
FPR

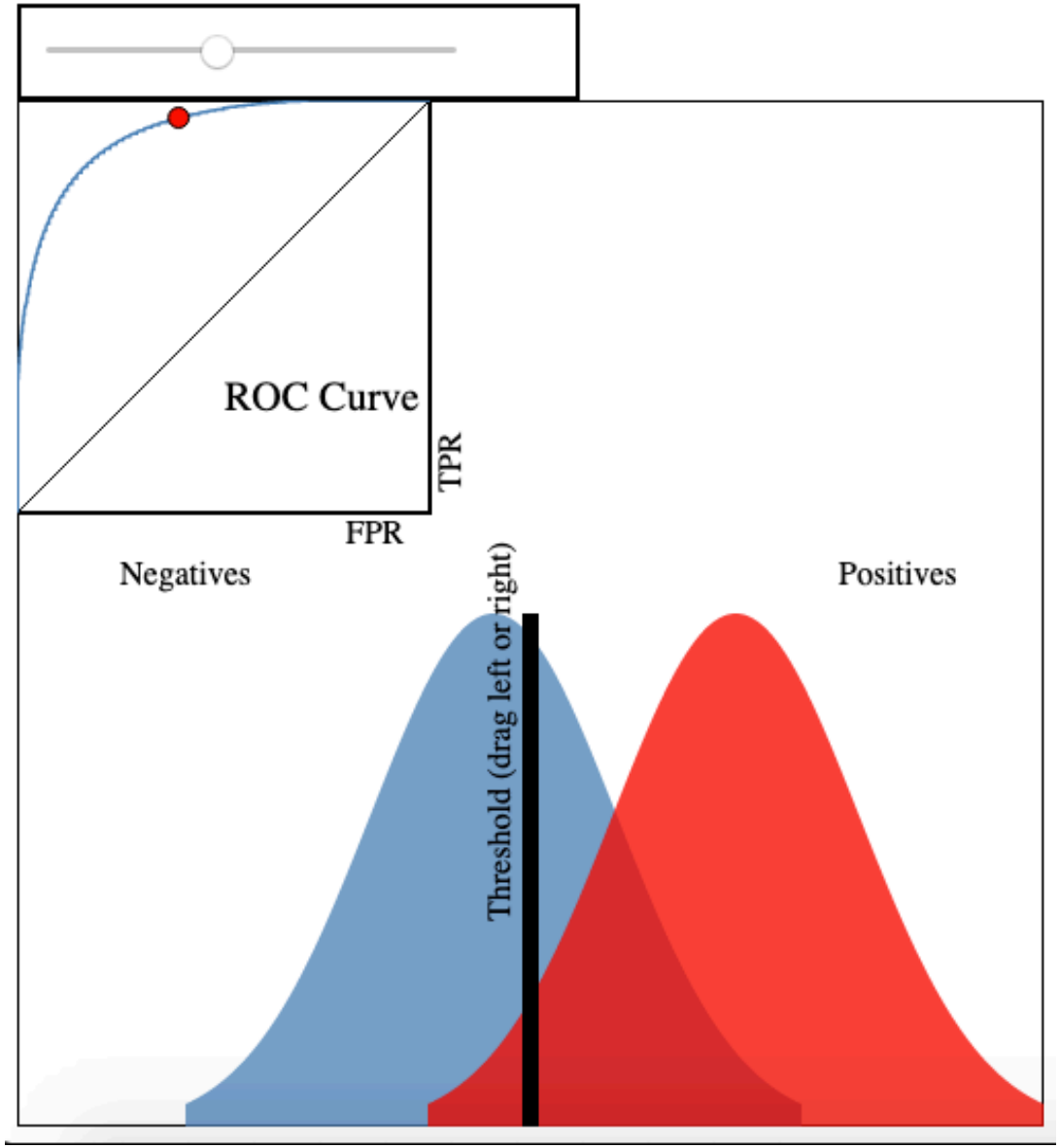

TN
TP
AUC = 1
0
0.5
1
Threshold


1
ROC
TPR
0
0
1
FPR


AUC = 0.5
0.5
Threshold


1
TPR
ROC
0
0
FPR
1

Visualization



http://www.navan.name/roc/

Replay Attacks

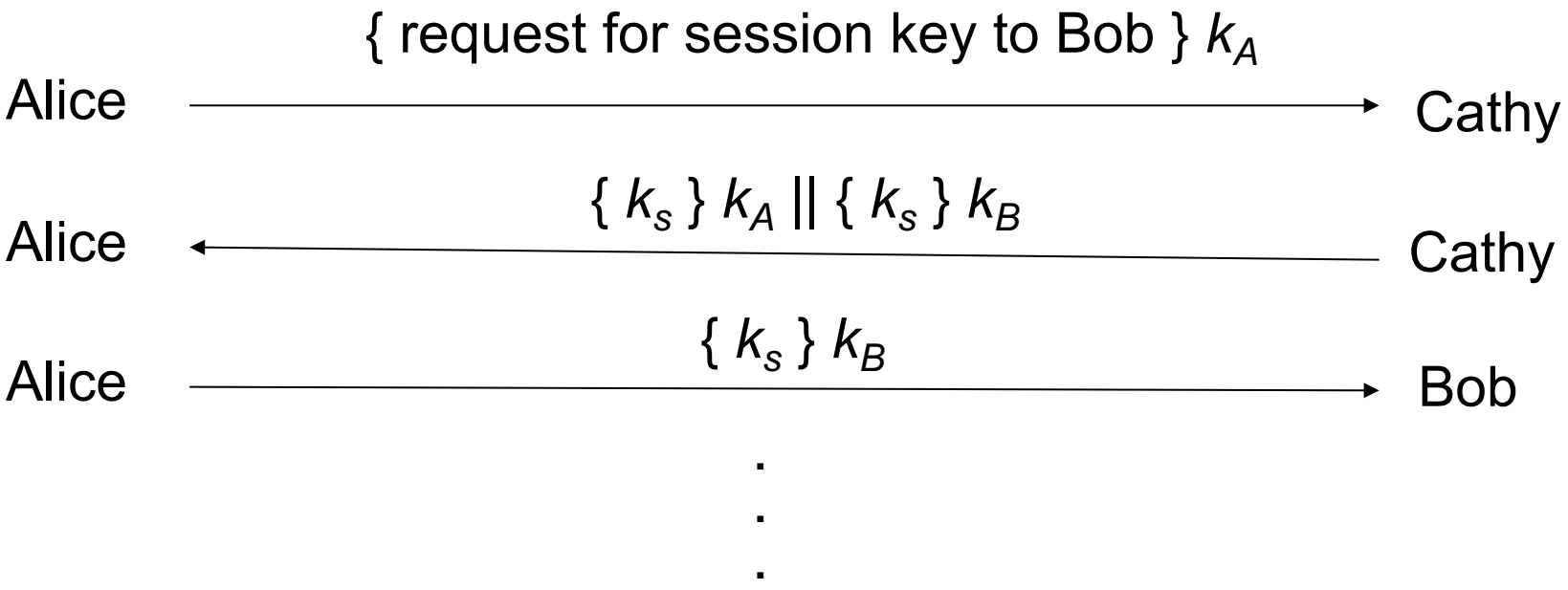
Replay Attacks

- Attacker doesn’t need your password
  - Can simply re-sent your encrypted password
- Big idea: attackers don’t need to know what they’re sending!
- Let’s look at 3 cases

Notation

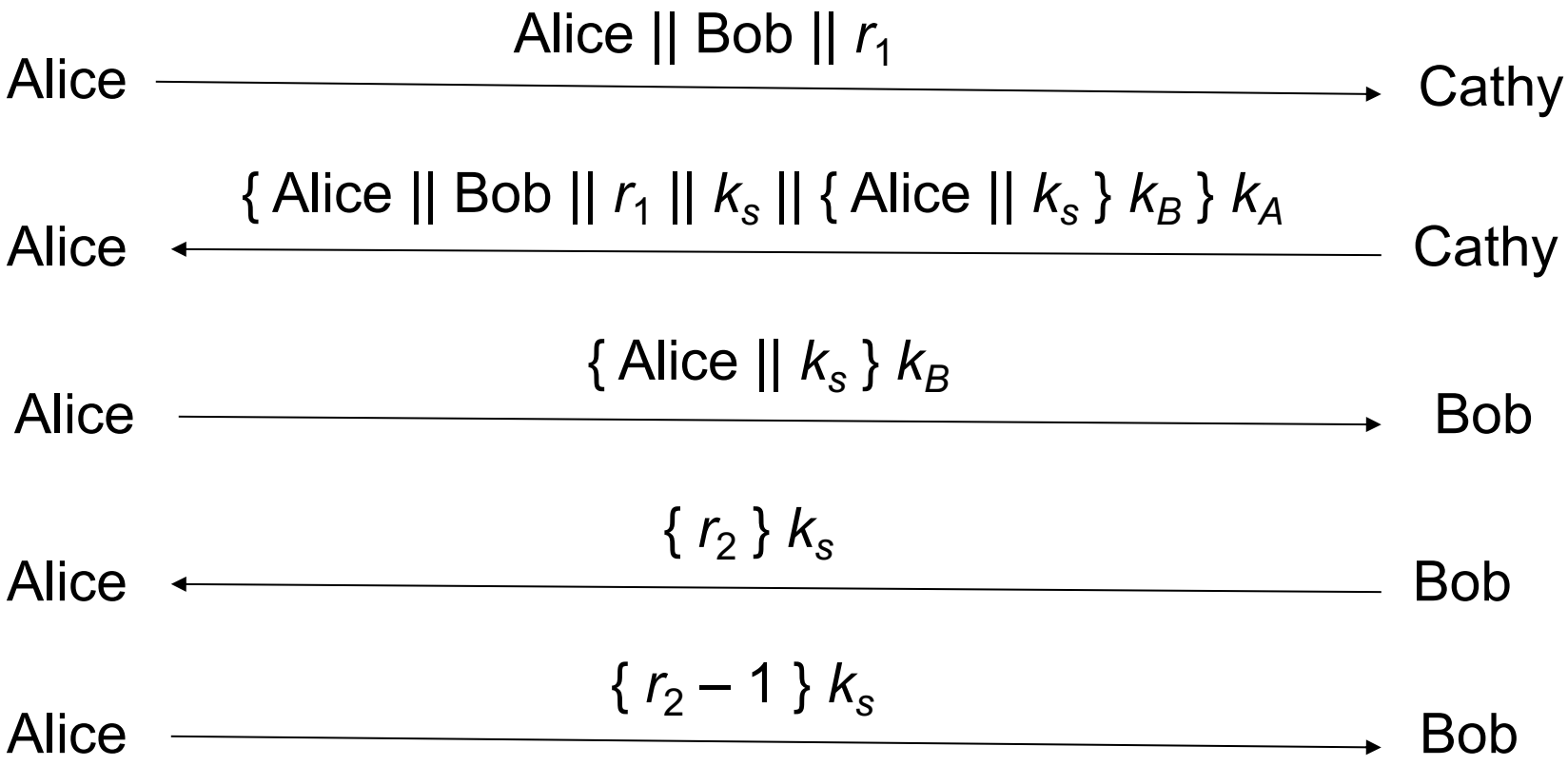
- $X \rightarrow Y : \{ Z \,||\, W \}\, k_{X,Y}$
  - X sends a message, Z concatenated with W, to Y
  - The message is encrypted with a shared secret key between X and Y
  - || is concatenation
  - $\{\}k$ is a message encrypted with k
- r_1, r_2 are nonces (nonrepeating random numbers)
- Alice wants to talk to Bob
  - Cathy is a trusted third party
- T is the time

Case 1

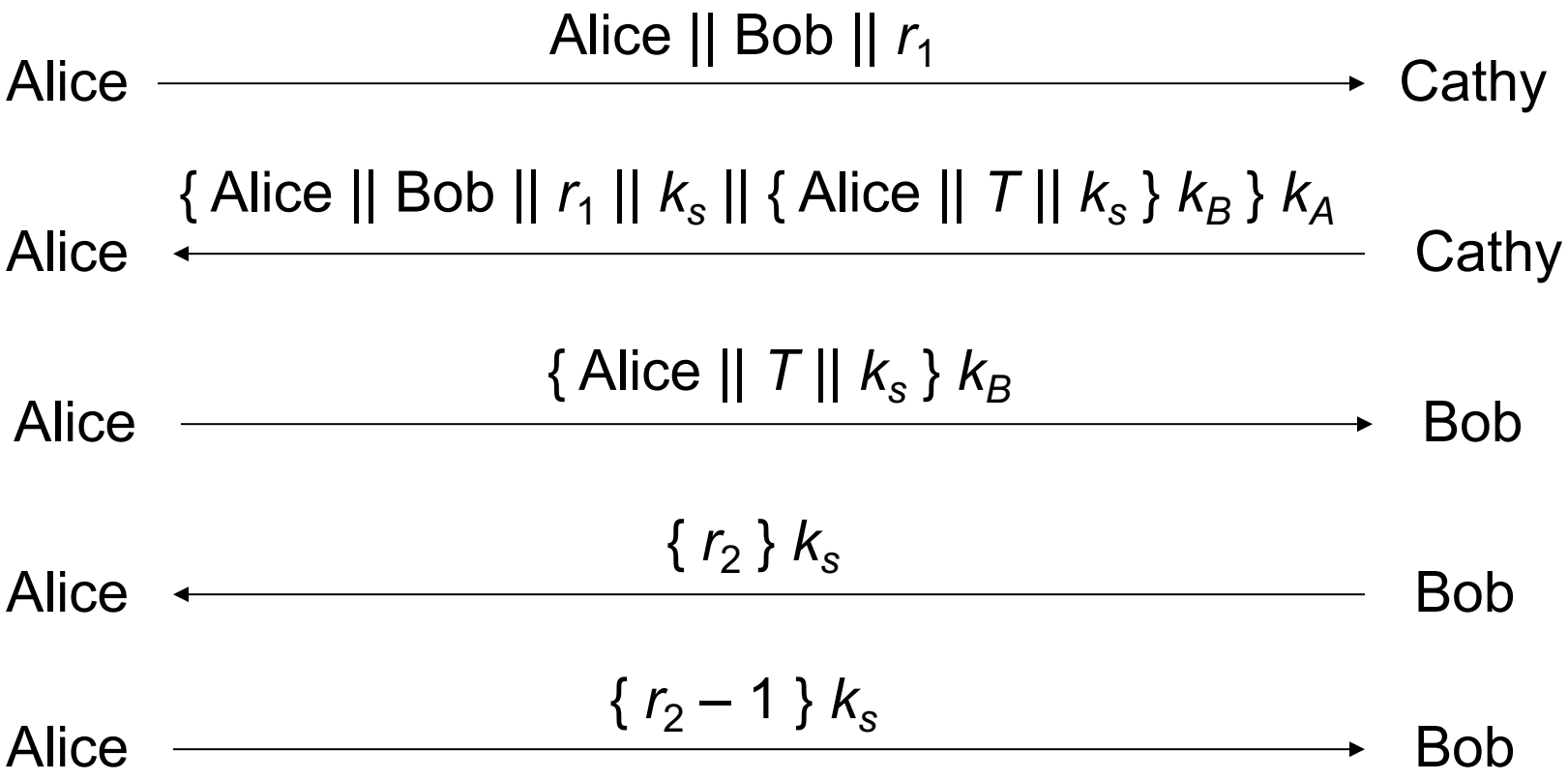


The rest of the conversation, encrypted with k_s

Case 2


Alice || Bob || r_1
Alice
Cathy
{ Alice || Bob || r_1 || k_s || { Alice || k_s } k_B } k_A
Alice
Cathy
{ Alice || k_s } k_B
Alice
Bob
{ r_2 } k_s
Alice
Bob
{ r_2 – 1 } k_s
Alice
Bob

Case 3


Alice || Bob || r_1
Alice
Cathy
{ Alice || Bob || r_1 || k_s || { Alice || T || k_s } k_B } k_A
Alice
Cathy
{ Alice || T || k_s } k_B
Alice
Bob
{ r_2 } k_s
Alice
Bob
{ r_2 – 1 } k_s
Alice
Bob

Takeaways

- Case 1 provides perfect confidentiality and integrity
  - But doesn't guarantee the message is sent currently!
- Case 2 is the Needham-Schroeder Protocol
  - Prevents replay attack
  - Still vulnerable if an attacker gets the session key
- Case 3 is the Needham-Schroeder Protocol with Denning-Sacco Modification
  - Fixes replay attacks, but we may have time issues
- I don't care if you know the names, but you may hear those terms

Kerberos

Kerberos

- Initially developed at MIT
- Authentication protocol
  - For both the client and server!
- “Ticket-based” Single-Sign On (SSO) protocol
- Most commonly seen today in Windows AD
  - Or Federated Identity Service!
- Based on Needham-Schroeder Protocol with Denning-Sacco modification

Kerberos

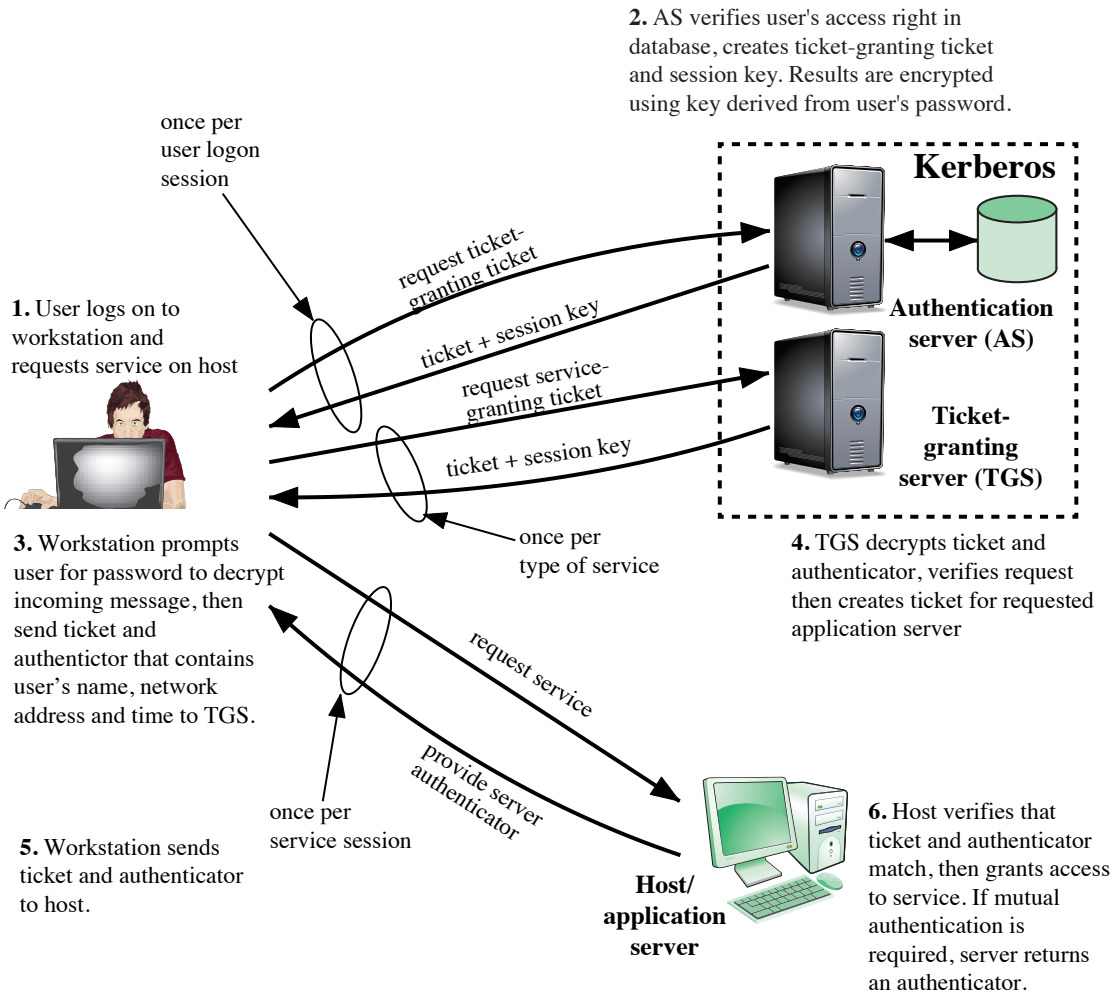


Figure 23.1 Overview of Kerberos

Realms

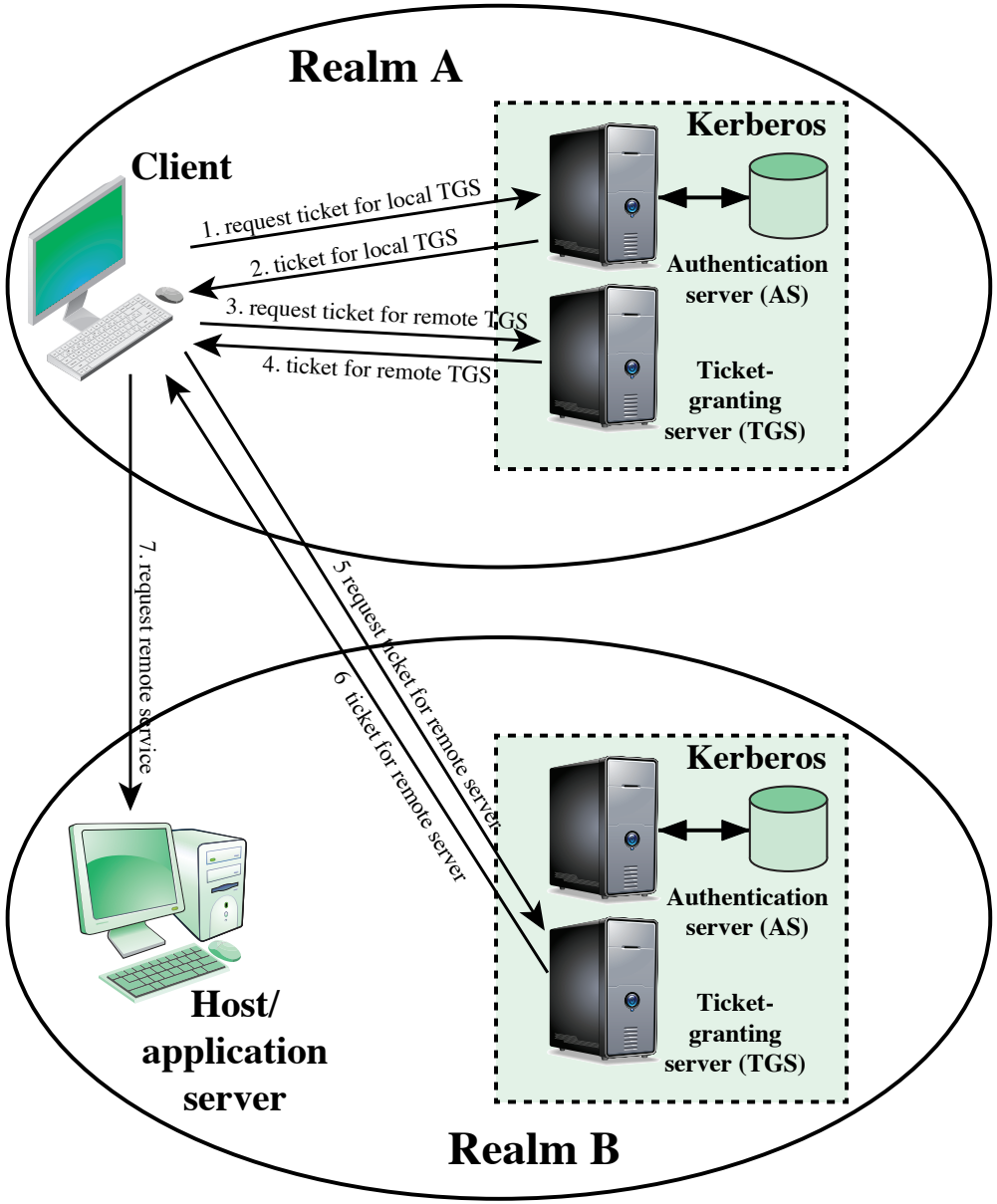


Figure 23.2 Request for Service in Another Realm

Kerberos Problems

- Generally seen as outdated
- Relies on having synchronized clocks
  - And caching exchanges for a certain amount of time
- Fixed field sizes leads to possible dictionary attacks

Basically...

Is Kerberos secure?

Kerberos is a network authentication protocol that is designed to provide strong authentication to client/ server applications by using secret-key cryptography. ... **Kerberos** is more **secure** than other authentication methods because it does not send plain text pass- words over the network and instead uses **encrypted** tickets.

www.pistolstar.com › pdfs › 18_Things_Kerberos

18 Things You Should Know About Kerberos Secure ... - PistolStar Inc.

www.scmagazineuk.com › article

'Devastating flaws' in Kerberos authentication protocol

Dec 14, 2015 - **Security** watchers warn of authentication and authorisation flaws in ... have surfaced relating to the **Kerberos** network **authentication protocol**.