CSE 40567 / 60567: Computer Security

Security Basics 4 / Cryptography 1
Homework #1 has been released. It is due Tuesday, Jan. 29th at 11:59PM

See **Assignments Page** on the course website for details
Course Roadmap

Basics
(weeks 1 & 2)

The Web
(weeks 15 & 16)

3 Core Areas
(weeks 3 - 6)

(weeks 6 - 10)

(weeks 11 - 15)
The history of computing and cryptography are intimately intertwined

• Turing served as a cryptanalyst at Bletchley Park during WWII
  ‣ Designed the electromechanical “Bombe” to decipher Enigma codes

• Colossus Mark 1
  ‣ First programmable, electronic, digital computer
  ‣ Designed to break the Lorenz cipher
Contemporary Cryptography

• Support security protocols that must operate in the presence of motivated attackers
  ‣ Hackers
  ‣ Criminals
  ‣ Corporations
  ‣ Governments

• Ensure that algorithms are themselves resistant to direct attack by cryptanalysis leveraging vast computational resources

• Design algorithms that run in realtime (even on embedded hardware)
What is the focus of this unit?

- The development of protocols that serve as the building blocks for system, network, web and mobile security
- Practical implementations and best practices for algorithms considered to be secure today
  - We’ll leave the proofs for CSE 40622/60622
- Real-world attacks, and how they can be mitigated
Introduction to Protocols
Eavesdropping revisited

Larger keyspaces supporting longer passwords and pin numbers are good, right?

128 bit key ➔ 4rch4304str0n0my
256 bit key ➔ m4ryh4d4l1ttl3l4mbl1ttl3l4mbwh0z

Doesn’t affect the “shoulder surfing” attack
Eavesdropping revisited

Master passwords based on a serial number provide a convenient fallback.

Serial numbers are rarely protected. (Mechanics, service technicians, janitors, etc. have access to them)
Eavesdropping revisited

What about a physical token?

Potential for replication if an attacker can gain access to it
Simple Authentication

Scenario: Alice wants to gain access to her workstation, but needs to authenticate via Bob.

\[ A \rightarrow B: A, \{A, N\}_{K_A} \]

- Alice
- Encryption
- Alice’s Key
- Bob
- Nonce
Nonce ($N$)

- One-time token
- Used to avoid a replay attack
Key diversification

Where does Alice’s key come from? One possibility:

Alice-specific identifier $\rightarrow$ Master Key

$K_A \rightarrow \{A\}_{K_M}$

Pros:
+ Simple key management

Cons:
- Length of identifier may limit usable keyspace
- Master key needs to be shared
Challenge-Response Protocols

• Problem with one-way authentication schemes: no guarantee messages make it to the intended recipient

• This can be solved with a two-way protocol
  1. Alice initiates an authentication session
  2. Bob responds with proof that he received Alice’s message
     ▸ Alice validates Bob’s message
Two-step challenge and response protocol

1. $A \rightarrow B: N$  
   \text{Shared Key}

2. $B \rightarrow A: \{B,N\}_K$

• In this scheme, Alice can decrypt the message from Bob, expecting to see the nonce she sent him.

• The shared key guarantees the integrity of the protocol.
  
  ‣ But how is the shared key distributed?
Two-factor Authentication

Let’s formalize two-factor authentication as a challenge-response protocol

$S = \text{Server}; \ P = \text{Password Generator};$

$PIN = \text{Personal Identification Number}$

1. $S \rightarrow A: N$
2. $A \rightarrow P: N, PIN$
3. $P \rightarrow A: \{N, PIN\}_K$
4. $A \rightarrow S: \{N, PIN\}_K$
Chip + Pin

Calculator uses bank card to perform crypto

1. Calculator is loaded with card
2. Asks for user’s PIN
3. For card transaction: computes response code based on a counter
4. For two-step logon: computes a challenge
How can two-factor authentication be attacked?

Phishing + Man-in-the-Middle
How can two-factor authentication be attacked?

1. Attacker installs Trojan program on Alice’s computer
2. When Alice logs into her bank, attacker piggybacks on that transaction with the Trojan

“…the horse which once Odysseus led up into the citadel as a thing of guile”

B. Schneier, “Two-Factor Authentication: Too Little, Too Late,” Schneier on Security, 2005
Defense against Man-in-the-Middle?

• For the banking scenario, derive the authentication code from:
  ‣ Transaction amount
  ‣ Payee account number
  ‣ Transaction sequence number

• This prevents an attacker from crafting their own transaction
Impact on usability

• Time-consuming: minutes instead of seconds
• Complicated: entry of a lot of information, including long strings of digits
  - Customers may revert to physical branches, call-centers and paper checks
    ▪ Loss of cost savings of online banking
Mutual Authentication

Alice and Bob need to identify each other:

1. $A \rightarrow B: N_A$
2. $B \rightarrow A: N_B$
3. $A \rightarrow B: \{N_B\}_K$
4. $B \rightarrow A: \{N_A\}_K$

What is the weakness in this protocol?
Reflection Attacks
Stopping reflection attacks

Alice and Bob need to identify each other; include IDs in the transaction:

1. $A \rightarrow B: N$
2. $B \rightarrow A: \{B, N\}_K$

- IDs are tied to a specific actor
- IDs can be checked with known actors
- If known actor didn’t send, reflection attack is detected
Manipulating the message

Alice $X_A$ Mallory $X_M$ Bob
Changing the environment

Original ATM

A Triton brand ATM with a dip style card reader and a triple DES keypad
BY-SA 3.0 Webaware

- End-to-end encryption

Switch to Cheaper ATM

- Doesn’t treat info on magnetic strip as secret
- Assumes operation in a trustworthy environment
Chosen Protocol Attack

Given some target protocol:
Design a new protocol that will attack the target protocol if users can be persuaded to reuse information

- Token
- Crypto Key
Chosen Protocol Attack

Protocols share elements

Protocol 2

Customer

Picture 143!

Prove your age by signing ‘X’

\[ \text{sig}_K \{X\} \]

Mafia porn site

Protocol 1

Buy 10 gold coins

Sign ‘X’

\[ \text{sig}_K \{X\} \]

BANK

Image credit: R. Anderson, Security Engineering
Ways to mitigate chosen protocol attack

- Do not allow crypto keys to be used by more than one application
- Do not let other people bootstrap their own application security off of yours
  - Be aware of security dependencies