6.1600 Recitation

September 2023

§1 Introduction and High Level Security Models

There are three things we need to consider when modeling any security system model

1. What is the system?
2. Who is the attacker?
3. How does the system and attacker interact?

Definition 1.1 (Kerckhoff's Principle)**.** We will always assume the attacker knows the implementation of whatever security mechanism we are using.

§2 Authentication Definition

Definition 2.1. Authentication refers to the process of verifying the identity of a user or process.

§3 Authenticating People and Attacks

§3.1 Passwords

This is the most basic form of authentication that we are all used to. However, a lot goes on within the server to ensure your password is safe.

Right after you register for an account...

1. The server generates a random *salt*.
2. Server will concatenate salt and password.
3. Then apply a hash function on the password and salt.
4. Finally, the output of the hash function $h = H(\text{salt}||\text{pw})$ and the salt will be stored in the database.

User	salt	$H(\text{salt}\|\|\text{pw})$
Alice	r_A	h_A
Bob	r_B	h_B
Charles	r_C	h_C

When a user later tries to login, server will take the inputted password, search the database for the salt, then compute $h = H(\text{salt}||\text{pw})$. If the hash matches, the user will be granted access.

Definition 3.1. A rainbow table is a precomputed table of hashed passwords.

The salting scheme talked about above defeats precomputation attacks.

§3.2 Request Authentication

Although password authentication is the people are most familiar with, a majority of authentication happens automatically. Often, clients will want to send an authenticated message to a server. That is, the client often wants to simultaneously authenticate to the server and send a request.

To do this, the client can send $(F(k, c||\text{req}), \text{req})$ to the server, where c is a challenge the server provides, and k is a key shared between the client and server.

§3.3 Phishing

Definition 3.2. A *phishing* attack is one where an attacker tricks a user into giving up their password.

These attacks are still very relevant today, but mechanisms such as TOTP codes can mitigate the attack.

§4 Hash Functions and their Properties

It is often useful to use a hash function during authentication schemes. We define a hash function as follows:

Definition 4.1. A hash function H maps a bitstring of any length onto a fixed-size space of inputs, as $H : \{0,1\}^* \to \{0,1\}^\lambda$.

§4.1 Collision Resistance

Definition 4.2. A hash function H is *collision resistant* if it is computationally infeasible to find two distinct strings x and x' such that $H(x) = H(x')$.

Another definition for collision resistant that may be more useful for proofs is

Definition 4.3. A function $H : \{0,1\}^* \to \{0,1\}^\lambda$ is collision-resistant if for all efficient adversaries $\mathcal{A}$, we have

$$\Pr[H(m_0) = H(m_1), m_0 \neq, _1 : (m_0, m_1) \leftarrow \mathcal{A}() \leq \text{negligible}.$$

In other words, the probability of finding a collision is so small that no efficient adversary can.

In practice, if the probability of collision is $< 2^{-128}$, we can be extremely sure that a collision will not occur.

§4.2 Merkle-Damgard

Merkle-Damgard construction is used to construct a large-domain CRHF from a small-domain CRHF

§4.3 Birthday Paradox

Given a hash function with λ-bit output, you can find a collision in time $O(\sqrt{2^\lambda})$. So if we want attacker to use at least 2^{128} bits to find a collision, our hash function must have at least 256 bits of output.

§5 Message Authentication Codes

MACs are used for authenticating communication. We assume both parties have access to a secret key k.

§5.1 MAC Security

A MAC over key space K and message space M is secure if any adversary wins the following game with negligible probability:

- Challenger generates a MAC key, k
- For n (polynomial n) iterations, the adversary sends any message m to the challenger
- The challenger responds with $MAC(k, m)$ for each of those messages
- The adversary sends (m^*, t^*)
- Adversary wins if m^* is not any previous message she sent, and $MAC(m^*, t^*) = t^*$

In other words, the adversary is able to generate a new message with a valid MAC for that new message.

§5.2 Pseudorandom Functions

MACs require pseudorandom functions to work (why?).

Definition 5.1. A pseudo random function is one that can be used to generate output from a random seed and a data variable, such that the output is computationally indistinguishable from truly random output

How do we generate pseudo random functions?

Definition 5.2. A function $f : \mathcal{X} \to \mathcal{Y}$ is a one-way function if for all efficient adversaries $\mathcal{A}$,

$$\Pr[f(\mathcal{A}(f(x))) = f(x) : x \leftarrow \mathcal{X}] \leq \text{negligible}.$$

In other words, suppose we have $f(x) = y$, then given x, y should be easy to compute, but given y and f we should not be able to easily compute x.

A subtle but important fact is: Psuedorandom functions exist if and only if one-way functions exist.

§5.3 In practice

In practice, we assume the following for one-way functions:

1. The function $f(x) := SHA256(x)$ is a one-way function where the domain is the set of 256-bit strings
2. The function $f(x) := AES(x, 0^{128})$ is a one-way function, where the domain is the set of 128-bit strings
3. The function $f(x) := 2^x \text{mod } p$ is a one=way function on domain $\{1, \ldots, p\}$ for large prime p

For PRFs

1. We typically use AES as a PRF. AES is a keyed function and takes key lengths of either $128, 192$, or 256 bits. Once keyed, it takes a 128 bit input and generates a 128 bit output. No mathematical proof exists that shows AES is a PRF, but it has undergone tremendous amounts of cryptoanalysis.

§5.4 Constructing Longer MACs

There are several methods, including

- CBC-MAC
  - Going out of favor, mostly because it is impossible to parallelize, so overhead is large
- Carter-Wegman MAC
  - Parallelizable
  - Uses the universal hash function, defined below.

Definition 5.3. H is a universal hash function if, for messages m and m', where $m \neq m'$, we have

$$\Pr[H(k, m) = H(k, m')] \leq \text{negligible}.$$

Weaker primitive than a CRHF. The adversary does not know the precise hash function that will be applied to their messages.

§6 Digital Signatures

What happens if a secret key cannot be shared ahead of time? Diffie-Hellman revolutionized the field with the introduction of public key cryptography. In essence, digital signatures work as follows:

- Suppose Alice is communicating with Bob
- Alice generates a signing key (sk) and verification key (vk)
- Alice makes her vk public (everyone in the world can see it)
- If Alice wants to send a message to Bob, she signs with her signing key
- Bob can use Alice's publicly posted verification key to verify the message came from Alice
- The point of digital signatures is that no one except Alice will be able to produce a signature on a message.

§6.1 Benefits of Digital Signatures

No longer need a channel to communicate, ease of access.

§6.2 Drawbacks of Digital Signatures

Large overhead. Typically what people do is use public key cryptography to exchange a secret key, then use symmetric key cryptography for the rest of the session.

§6.3 How do you Construct Digital Signatures?

- Lamport's Scheme for one time secure signatures
- Extend to arbitrary-length messages
- Then go from one-time secure to many-time secure. This uses a construction similar to Merkel tree

More details of the exact implementation/proofs are in the book.

§6.4 Public Key Infrastructure

How do we actually verify Alice's verification key is what she says it is?

- Use verification key as identities
  - Hard to remember, no way to update name-to-key mapping
  - However, some services such as Bitcoin does this – you'd send money to an account identified by their public key.
- Trust on first use. Client accepts the first verification key it sees associated with such name, and later on the client will only accept the same verification key for that name
  - If the first key client receives is incorrect, it's game over. How do we handle key updates?
  - Some services such as ssh uses this
- Certificates. A website obtains a certificate from a certificate authority (CA) to verify its public key. When clients connect to the website, they receive and validate this certificate, ensuring it's CA-signed.
  - If any CA gets compromised, it'll be really bad. How would revoking certificates work? CAs don't often perform thorough investigation of validation.
  - However, in practice this works very well and is quite often used.