CSE 127 Computer Security
Alex Gantman, Spring 2018, Lecture 8

Cryptography II: PKI, Protocols, Side Channels
Review
Using Cryptography

- Alice wants to send (a plaintext) $m$ to Bob, via a channel that is controlled by Eve
Cryptographic Primitives

- Confidentiality
  - Symmetric Encryption
    - $c = E_k(m), m = D_k(c)$
  - Asymmetric Encryption
    - $c = E_K(m), m = D_k(c)$
  - Combining Asymmetric with Symmetric
    - $k' \leftarrow r, E_K(k') || E_k(m)$
  - You can rely on plaintext remaining secret.
    - Ciphertext reveals nothing about plaintext contents
  - You **cannot** rely on plaintext remaining unmodified.

- Integrity and Authenticity
  - Symmetric MAC
    - $a = MAC_k(m)$
  - Asymmetric Signature
    - $s = S_k(H(m))$
    - $V_K(s, H(m))$: returns true or false
  - You can rely that whoever generated the tag (MAC or signature) had the secret key.
  - You **cannot** rely on tag not leaking information about the message.
Cryptographic Primitives

- **Authenticated encryption** simultaneously provides confidentiality, integrity, and authenticity.

- A **cryptographic hash function** maps arbitrary length input into a fixed-size string and has the following properties:
  - **Pre-image Resistance**
    - Impractical to find an input (pre-image) that generates specified output.
  - **Collision Resistance**
    - Impractical to find two inputs that hash to the same output.
  - Can be used to **commit** to a value.
  - Can be used as a fixed-size **fingerprint** of a longer string.
Using Cryptography

- Alice wants to send (a plaintext) $m$ to Bob, via a channel that is controlled by Eve.
- Alice and Bob know each other’s public keys.
- Alice and Bob establish a secure “pipe”.
- Eve cannot see plaintext contents inside the pipe, or modify them without detection.
Using Cryptography

- Alice and Bob got secrecy + integrity + authenticity and everyone lived happily ever after, right?
- Let’s try to understand exactly what we achieved
Using Cryptography

- Bob knows that
  - Alice (or someone with her private key) knows the plaintext
  - Alice (or someone with her private key) signed the plaintext at some point in the past

- Alice knows that
  - Only Bob (or someone with his private key) can read the plaintext
  - Bob (or anyone that gets the plaintext and Alice’s signature) can prove that Alice signed the plaintext
Using Cryptography

- Alice does not know:
  - Whether Bob receives the message
  - When Bob receives the message
  - How many times Bob receives the message
  - Whether Bob keeps the message secret

- Bob does not know:
  - Did Alice address this message to Bob
  - Who sent this copy of the message
  - When the message was sent
  - Who else knows the plaintext
Digital Signatures

▪ What Does Signing Mean?
  – Signing is a mechanical operation that has no meaning in itself.

▪ What cryptography promises:
  – Only someone who knows the private key can create a signature that verifies using the corresponding public key

▪ Meaning of a digital signature is a matter of convention
  – Code signing: signer attests software is authorized to be installed
  – Email signing: signer attests she wrote message
  – Certificate signing: (coming up next!)

▪ Both signer and verifier should agree on meaning
Public Key Infrastructure (PKI)
Using Cryptography

- Alice wants to send (a plaintext) $m$ to Bob, via a channel that is controlled by Eve.
- Alice and Bob know each other’s public keys.
- Alice and Bob establish a secure “pipe”.

Asymmetric Cryptography

- Public directory contains everyone’s public key
- To encrypt to a person, get their public key from directory
- No need for shared secrets!
Getting Public Keys

- Alice and Bob need a way to get each other’s public key.
- Alice can send an unencrypted message to Bob:
  - “Hey, send me your public key. Here’s mine.”
- Bob sends Alice his public key.
- They communicate securely ever after?
Getting Public Keys

- What they want to happen

Alice $\to$ Bob $K_A$
Bob $\to$ Alice $K_B$

- What happens instead

Alice $\to$ Eve $K_A$
Eve $\to$ Alice $K'_B$
Eve $\to$ Bob $K'_A$
Bob $\to$ Eve $K_B$
Getting Public Keys

- If Eve has man-in-the-middle capability, she can impersonate Alice to Bob and Bob to Alice.
  - Eve becomes invisible gateway between them.
  - Alice and Bob have no idea Eve is there.
Getting Public Keys

- Alice and Bob need a way to know that each has the real public key of the other.
- Ideal solution: Alice and Bob meet in person and exchange public keys
- Equivalent: Alice and Bob meet in person and exchange public key fingerprints
  - Key fingerprint: cryptographic hash of public key
  - Public key itself can be sent in the open
Getting Public Keys

- Problem with ideal:
  - We are back to pair-wise key establishment
  - Alice and Bob need to meet
  - Impractical to meet and verify key of everyone you talk to

- Any security problem can be solved with a trusted third party
Getting Public Keys

- Using a trusted intermediary
  - Alice and Bob have already exchanged keys with Charlie
  - Charlie sends signed message with Alice’s key to Bob
  - Charlie sends signed message with Bob’s key to Alice
  - Alice and Bob trust Charlie to send the real public keys
  - Alice and Bob now have each other’s public key
Getting Public Keys

- We can do better...
- Charlie creates a **certificate**:
  - “I, Charlie, verified that Alice’s key is ...”
- Charlie signs the message and gives it to Alice
  - Alice now has certificate attesting to her public key
- Alice sends Bob her public key and Charlie’s certificate
- Bob verifies signature on certificate
- Bob trusts Charlie, accepts public key from Alice
Who is Charlie?

- Two common models:
  - PGP: Charlie is any other person you trust.
  - Almost everywhere else: Charlie is a *Certificate Authority*. 
Pretty Good Privacy (PGP) is an application (and associated protocols) used for signing, encrypting, and decrypting texts, e-mails, files, directories, etc.

PGP allows one user to attest to the accuracy of another user’s public key — **key signing**
- PGP does not use the term “certificate”
- Public key has set of attestation signatures (certificates)

A user can indicate how much she trusts another user’s signature on a key.
PGP Web of Trust

- Alice’s signature on Bob’s PGP key means Alice has verified that this is really Bob’s key
  - Email address and name associated with key are really his

- Other people who trust Alice can use her signature on Bob’s key to be sure it is Bob’s key
Certificate Authorities

▪ An alternative to PGP-like web of trust is to rely on centralized **Certificate Authorities**: trusted signers of public keys.

▪ CA model used to sign certificates used on Web.

▪ Your browser has a set of public keys of trusted CAs.
  – Who makes this list?
  – How many CAs are on the list?
  – Who are these CAs?

[UC San Diego]
## Certificate Authorities

<table>
<thead>
<tr>
<th>Certificate Authority</th>
<th>Licensee or Holder Name</th>
<th>Organization Name</th>
<th>Organization Country</th>
<th>Organization Province</th>
<th>Valid From</th>
<th>Valid To</th>
<th>Organization Algorithm</th>
<th>Organization State</th>
<th>Organization Country</th>
<th>Organization State</th>
<th>Organization Country</th>
<th>Organization State</th>
<th>Pref. Domain</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>Apple</td>
<td>Apple Computer Co., Ltd.</td>
<td>CA-Apple</td>
<td>United States</td>
<td>California</td>
<td>2016-01-01</td>
<td>2021-01-01</td>
<td>RSA</td>
<td>CA</td>
<td>United States</td>
<td>California</td>
<td>United States</td>
<td>California</td>
<td>1440</td>
<td>1</td>
</tr>
<tr>
<td>Akamai</td>
<td>Akamai Technologies</td>
<td>CA-Akamai</td>
<td>United States</td>
<td>California</td>
<td>2016-01-01</td>
<td>2021-01-01</td>
<td>RSA</td>
<td>CA</td>
<td>United States</td>
<td>California</td>
<td>United States</td>
<td>California</td>
<td>1440</td>
<td>1</td>
</tr>
<tr>
<td>Amazon</td>
<td>Amazon Web Services</td>
<td>CA-Amazon</td>
<td>United States</td>
<td>California</td>
<td>2016-01-01</td>
<td>2021-01-01</td>
<td>RSA</td>
<td>CA</td>
<td>United States</td>
<td>California</td>
<td>United States</td>
<td>California</td>
<td>1440</td>
<td>1</td>
</tr>
<tr>
<td>Akamai</td>
<td>Akamai Technologies</td>
<td>CA-Akamai</td>
<td>United States</td>
<td>California</td>
<td>2016-01-01</td>
<td>2021-01-01</td>
<td>RSA</td>
<td>CA</td>
<td>United States</td>
<td>California</td>
<td>United States</td>
<td>California</td>
<td>1440</td>
<td>1</td>
</tr>
<tr>
<td>Amazon</td>
<td>Amazon Web Services</td>
<td>CA-Amazon</td>
<td>United States</td>
<td>California</td>
<td>2016-01-01</td>
<td>2021-01-01</td>
<td>RSA</td>
<td>CA</td>
<td>United States</td>
<td>California</td>
<td>United States</td>
<td>California</td>
<td>1440</td>
<td>1</td>
</tr>
</tbody>
</table>

**Count:** 153
Certificate Authorities

- Mozilla
  - ~150 root certificates
  - [https://wiki.mozilla.org/CA/Included_Certificates](https://wiki.mozilla.org/CA/Included_Certificates)

- iOS
  - ~150 root certificates

- Microsoft
  - ~300 active root certificates
  - [http://aka.ms/RootCert](http://aka.ms/RootCert)
Certificate Authorities

- Certificate semantics:
  - Subject (name, domain)
  - Issuing CA
  - Validity period
  - Limitations on use
Certificate Authorities

- Which CA can issue a certificate for mycompany.com?
- For fbi.gov?
Let’s Encrypt

Web Server
Admin Software

example.com

Put ed98 at https://example.com/8303
Sign 9cf0b331

https://letsencrypt.org/how-it-works/
Let’s Encrypt

https://letsencrypt.org/how-it-works/
Certificate Authorities

- What if we take a **Trusted Third Party** and combine it with **Another Layer of Indirection**?

- **Certificate Hierarchy**

- **Root CA** signs keys for **Intermediate CAs**, which in turn sign keys for users (or other intermediate CAs)
Certificate Authorities

- Certificate hierarchy for ucsd.edu
Certificate Authorities

- Certificates also used in code signing
  - https://source.android.com/security/apksigning/
  - https://docs.microsoft.com/en-us/windows-hardware/drivers/install-driver-signing

- Who is the CA?

- What is the meaning of the signature?
  - Alice released this app?
  - Alice authorizes this app to run?
  - Alice authorizes this app to access privileged resources?
Certificate Revocation

▪ What happens if someone steals your private key?
  – They can impersonate you and read messages encrypted to you

▪ Certificate expiration helps with this but not enough

▪ CA and PGP PKIs support revocation
  – Owner says: “I, Alice, revoke my public key ... do not use it.”
  – Signs revocation with her private key
  – Others can verify Alice’s signature, stop using key
Certificate Revocation

- How does Bob know if Alice’s key has been revoked?
- Bob asks Alice: “Has your key been revoked?”
- Alice sends signed message: “No.”
- Does not work: if Alice’s key has been compromised, then Eve could have forged the message “No.”
- Availability of revocation list critical
Certificate Revocation

- In PGP model, only Alice can revoke her own key
  - If Alice loses her private key, she can’t revoke
  - Do not lose private key
  - Option: generate revocation with key, store in secure place

- In CA model, Alice asks CA to revoke certificate
  - Alice does not need private key to do this, can authenticate herself through other means
Certificate Revocation

- Two Mechanisms: CRL and OCSP

- **Certificate Revocation List (CRL):**
  - Certificate says where to get CRL
  - Clients periodically download updated CRLs
  - What if CRL server is down?
Certificate Revocation

- Two Mechanisms: CRL and OCSP

- **Online Certificate Status Protocol (OCSP):**
  - Query CA about status of cert before trusting it
  - “You said I can trust this key, but are you still sure?”

- OCSP Stapling
  - Server includes recent OCSP status (signed by CA)

- Aside: Certificate Pinning
  - Remember which certificate was used for a particular domain and raise an alert if a different one is used later

- Visit [https://revoked-isrgrootx1.letsencrypt.org/](https://revoked-isrgrootx1.letsencrypt.org/) with your browser
Protocol Examples
Secure Shell (SSH)

- "Secure Shell (SSH) provides a secure channel over an unsecured network, connecting an SSH client application with an SSH server. Common applications include remote command-line login and remote command execution, but any network service can be secured with SSH."

1. Client initiates the connection by contacting server
2. Sends server public key
3. Negotiate parameters and open secure channel
4. User login to server host operating system

https://www.ssh.com/ssh/
Secure Shell (SSH)

- No trusted authorities
- Certificate pinning
Transport Layer Security (TLS)

- “When secured by TLS, connections between a client (e.g., a web browser) and a server (e.g., wikipedia.org) have one or more of the following properties:
  - The connection is private (or secure) because **symmetric cryptography** is used **to encrypt** the data transmitted...
  - The identity of the communicating parties can be **authenticated using public-key cryptography**...
  - The connection ensures integrity because each message transmitted includes a **message authentication code** to prevent undetected loss or alteration of the data during transmission.”

- Supersedes obsolete SSL (Secure Socket Layer)
Certificate Authorities

- HTTPS is secured by TLS
Certificate Authorities

- HTTPS is secured by TLS
Content Delivery Networks (CDNs)

- CDN: geographically distributed network of proxy servers
  - Cache static content closer to the requester
  - Improve latency
  - Decrease network congestion
  - Improve reliability and availability
    - DDOS protection

- Mess up our nice security abstractions
  - Now Alice deliberately wants her CDN to impersonate her to Bob!
Content Delivery Networks (CDNs)

- Bob wants to connect to www.fbi.gov
- Bob’s browser attempts to get the corresponding IP address via DNS
- Because FBI is using Cloudflare CDN, DNS resolves to a Cloudflare edge server
- But Bob’s browser thinks it’s talking to fbi.gov
- Cloudflare needs to convince Bob’s browser that it’s really FBI
Content Delivery Networks (CDNs)

- Deputized via “Subject Alternate Name” field
  - “Yeah, I’m cloudflaressl.com, but I’m authorized to communicate on behalf fbi.gov”

- Who decides whether a CDN can get a given Subject Alternate Name in its cert?
Side Channels
Side Channels

- We often think of systems as black boxes:
  - As abstractions that consume input and produce output.

- Sometimes, in addition to what is in the output, critical information can be revealed in *how* it is produced.
  - This becomes a *side channel*: a source of information beyond the output specified by the abstraction.
Side Channels

- Encryption example
  - Adversary does not gain any information about the plaintext from seeing the ciphertext.
  - Abstraction: adversary can observe output, but encryption function works like a black box.
  \[ c = E_k(m) \]
  - Reality: encryption function performs operations that may depend on values of the key and the plaintext

- What side channel information can we get?
  - How long does the computation take?
  - How much power does it take?
  - Which errors/exceptions/debugging output does it produce?
  - Etc.
Consumption Side Channels

- How long does this password check take?
  - Depends on where the first mismatch is.

```c
char pwd[] = “z2n34uzbnqhw4i”;
//...
int CheckPassword(char *buf)
{
    return strcmp(buf, pwd);
}
```
Side Channels

- Consumption: how much of a resource is being utilized to perform the operation?
  - Examples: time, power, memory, network, etc.

- Emission: what out-of-band signal is generated in the course of performing the operation?
  - Examples: electro-magnetic radiation, sound, movement, error messages, etc.
Side Channel Examples

- Tenex password verification
  - Alan Bell, 1974
  - Character-at-a-time comparison + virtual memory
  - Recover the full password in linear time

https://www.sjoerdlangkemper.nl/2016/11/01/tenex-password-bug/
Side Channel Examples

- Simple Power Analysis (SPA)
- Differential Power Analysis (DPA)
  - Paul Kocher, 1999
  - Using signal processing techniques on a very large number of samples, iteratively test hypothesis about secret key bit values.
Side Channel Examples

- **Timing Analysis of Keystrokes and Timing Attacks on SSH**
  - Dawn Song, David Wagner, Xuqing Tian, 2001
  - Recover characters typed over SSH by observing packet timing

Figure 1: The traffic signature associated with running `SU` in a SSH session. The numbers in the figure are the size (in bytes) of the corresponding packet payloads.
Side Channel Examples

- **Remote Timing Attacks are Practical**
  - David Brumley, Dan Boneh, 2002
  - Recover RSA private keys by observing packet timing
  - Typical attack takes approximately 2 hours (1433600 queries)
Side Channel Examples

- **Side-Channel Leaks in Web Applications: a Reality Today, a Challenge Tomorrow**
  - Shuo Chen, Rui Wang, XiaoFeng Wang, Kehuan Zhang, 2010
  - Recover characters typed into search boxes over HTTPS by observing sizes of responses

Figure 2: User interface for adding health records

Side Channel Examples

- **Keyboard Acoustic Emanations**
  - D. Asonov, R. Agrawal, 2004
  - Recover keys typed by their sound

- **Keyboard Acoustic Emanations Revisited**
  - Li Zhuang, Feng Zhou, J. D. Tygar, 2009

[Graph showing relationship between length of recording and recognition rate.]

Fig. 7. Length of recording vs. recognition rate.

Side Channel Examples

- **Breaking the x86 ISA**
  - Christopher Domas, 2017
  - Reverse engineer undocumented instructions

https://github.com/xoreaxeae/sandsifter
Side Channel Examples

- And countless others...
Side Channel Attacks

- **Covert Channels**
  - When the sender is malicious and is deliberately using a side channel
  - What if Alice is deliberately trying to leak information to Eve
Mitigating Side Channels

▪ Eliminate dependency on secret data

▪ Make everything the same
  – Use the same of amount of resources every time
  – Hard (too many optimizations in hardware, compilers, etc.)
  – Expensive (everything runs at worst-case performance)

▪ Hide
  – “Blinding” can be applied to input for some algorithms
  – Only possible for certain algorithms (eg RSA)

▪ Adding random noise?
  – Attacker just needs more measurements to extract signal
Side Channels

- Faults can create additional side channels or amplify existing ones
  - Erroneous bit flips during secret operations may make it easier to recover secret internal state

- Attackers can induce faults, leading to \textit{fault injection attacks}
  - Glitch power, voltage, clock
  - Vary temperature
  - Subject to light, EM radiation
Fault Attacks

- **On the Importance of Checking Cryptographic Protocols for Faults**
  - "We present a theoretical model for breaking various cryptographic schemes by taking advantage of random hardware faults... An implementation of RSA based on the Chinese Remainder Theorem can be broken using a single erroneous signature. Other implementations can be broken using a larger number of erroneous signatures..."
Fault Attacks

- **Bug Attacks**
  - Eli Biham, Yaniv Carmeli, and Adi Shamir, 2008
  - “In this paper we show that if some intelligence organization discovers (or secretly plants) even one pair of single-word integers $a$ and $b$ whose product is computed incorrectly (even in a single low order bit) by a popular microprocessor, then any key in any RSA-based security program running on any one of the millions of PC’s that contain this microprocessor can be easily broken... In some cases, the full key can be retrieved with a single chosen ciphertext...”
Side Channels

- The Pentagon Press Parking Index

I hesitate to post the Pentagon Press Parking Index (PPPI) tonight because if it's flashing red tomorrow, I may not share. But for the next 12 hours, prospects of Assad strikes are low.
Homework

- Project 3 is due 4/30 @ 10pm
- Midterm is on 5/3
Next Lecture...

User Authentication