Fundamental Concepts

Historically, computer security tried to fulfill following three principles: C.I.A.

- **Confidentiality**
  It is about protecting data so that those who are allowed to see it can and those who are not allowed to see it won’t.

  *What types of data might someone want to keep secret?*

  *Some tools to maintain confidentiality:*
  (i) Encryption
  (ii) Access control
  (iii) Physical security

- **Integrity**
  It is about protecting data so that it does not get altered in any unauthorized way. Notice that it’s possible for data to be kept confidential but for its integrity to be compromised. *How?*

  *Some detection and correction tools:*
  (i) Checksums
  (ii) Data correcting codes
(iii) Back-ups

- **Availability**
  It is about making information (or a resource) accessible and modifiable in a timely fashion to those who are authorized to do so. Again, data can be kept confidential and unaltered but it not useful if it cannot be accessed.

  Computational redundancies to ensure availability:

Three other concepts important to modern computer security: **A.A.A.**

- **Assurance**
  It refers to how trust is provided and managed in computer systems. It involves
  
  policies
  
  permissions
  
  protections

- **Authenticity**
  Authenticity can refer to the validation of a person or system’s identity or it can refer to the determination that the statements issued by a person or a system is genuine.

  To verify a person’s identity, we can use

  *Non-repudiation* is the property that parties involved in a contract will fulfill their stated responsibilities. That is, they cannot deny that they signed the contract. The most common tool for achieving this is
• **Anonymity**

It is the property that certain records or transactions cannot be attributed to a particular individual.

*Some tools for achieving anonymity:*

(i) Aggregation

(ii) Proxies

(iii) Pseudonyms

**Exercise:** The hacking of Mat Honan’s digital life. Which of C.I.A. and A.A.A. were compromised? How?
Common Threats and Attacks

1. *Eavesdropping* – intercepting information intended for someone else

2. *Alteration* – unauthorized modification of information

3. *Denial-of-service* – interrupting or degrading access to some information or service

4. *Masquerading* – pretending to be someone else

5. *Repudiation* – denying a commitment or receipt of data

6. *Correlation and traceback* – integrating multiple data sources and information flows to determine the source of a particular piece of information
Cryptographic Concepts

This discussion is just an overview of cryptography. We will study it in more detail later.

Encryption. In encryption, the traditional goal is *confidentiality*. Alice wants to send a message $M$ to Bob. Unfortunately, their channel of communication is insecure – another person can eavesdrop on their conversation. What should Alice do?

*The solution.*

The encryption and decryption processes are conducted in such a way that even if someone intercepts the ciphertext $C$, it will be infeasible for that person to determine the plaintext $M$.

In practice, $E$ and $D$ are well-known algorithms. To make $M$ hard to decipher, Alice and Bob makes use of *keys*.
Formally, a cryptosystem consists of the following components:

- the set of plaintexts
- the set of ciphertexts
- the set of encryption keys
- the set of decryption keys
- the correspondence between the encryption and decryption keys
- the encryption algorithm
- the decryption algorithm

Example: The shift cipher. In this cryptosystem, each plaintext character $c$ is replaced by another character that is $k$ circular shifts from it. When $k = 3$, this cipher is referred to as Caesar cipher because Julius Caesar (100 BC - 44 BC) apparently used it to communicate with his generals. Here’s an example of the Caesar cipher:

Plaintext: LEAVE NOW
Ciphertext:

Let us now formalize the shift cipher.

- the set of plaintexts
- the set of ciphertexts
- the set of encryption keys
- the set of decryption keys
- the correspondence between the encryption and decryption keys
- the encryption algorithm
- the decryption algorithm
Modern Cryptosystems. Modern cryptosystems are significantly more complicated than the shift cipher. They are often classified into two groups:

1. Symmetric (key) Encryption

   The term symmetric is used here because the encryption key used by Alice and the decryption key used by Bob are either identical or simple transformations of each other. In other words, Alice and Bob share a key.

   • Examples:

   • Speed:

   • Key Distribution:

2. Public key Encryption

   Suppose Alice wants to communicate with Bob. Bob has two keys: a private key $S_B$ which he keeps secret, and a public key $P_B$ which he advertises to the public. Alice uses Bob’s public key $P_B$ for encryption; Bob uses his private key $S_B$ for decryption.
• Examples:

• Speed:

• Key Distribution:

Final note: In practice, because of the speed differences between the two methods, public-key encryption is typically used for exchanging a shared key between Alice and Bob. They would then use symmetric encryption for the rest of their communication.
Digital Signatures

Digital signatures are electronic versions of the ink signatures. If Bob digitally signs a statement $M$ with signature $S$, a judge must be able to verify that $S$ is indeed Bob’s signature and not somebody else’s. Interestingly, public-key cryptosystems can be used for digital signatures.

Let $E$ and $D$ be the encryption and decryption algorithms of a public-key cryptosystem. Recall that $S_B$ and $P_B$ are Bob’s private and public keys. Given a message (or statement) $M$, the following is true:

It turns out that the following is true as well:

Here’s the procedure for signing $M$:

- Bob signs $M$:
- Bob attaches his signature to $S$ to $M$ so that the “document” is now $(M, S)$.
- If a judge wants to verify that $S$ is Bob’s signature, she will do the following:

Why couldn’t someone else have forged $S$?
Cryptographic hash functions

A cryptographic hash function $h$ takes a message $M$ (which can be of arbitrary length) and returns a hash value $h(M)$ of a fixed length. The value $h(M)$ is also referred to as the message digest of $M$, the fingerprint of $M$ or the checksum of $M$. Examples of cryptographic hash functions used in practice include the MD5 which returns a value with 128 bits, SHA-1 which returns a value of 160 bits, and SHA-256 which returns a value of 256 bits. Here’s are the hash values for the word “Packers”.

- MD5(Packers) = bd2905995cb9f80979096f1ce855b269
- SHA-1(Packers) = a88fa70b3bde7c4b9ef5a06e9950dab71b22858c
- SHA-256(Packers) = c2e6d3a292f26c7dad371128982a076fa9034f4c71fc0b43d9ecce741e11183

Desirable properties of cryptographic hash functions:

1. one way:

   In general, a little change in $M$ should create a big change in $h(M)$.

   - MD5(Packers) = bd2905995cb9f80979096f1ce855b269
   - MD5(Pack3rs) = d09e37b76f3b3f8c7c679324b99b1232

2. collision-resistant

3. strong collision-resistant*

Applications:

1. File system integrity
2. Digital signatures

3. Message integrity