## Some basic defns

- Cryptography: the study of mathematical techniques related to information security that have the following objectives:
  - **Authentication**: corroboration of the identity of an entity.
  - **Confidentiality**: ensuring information is accessible only by authorized persons.
  - **Integrity**: ensuring information is has not been altered by unauthorized or unknown means.
  - **Non-repudiation**: preventing the denial of previous commitments or actions.

## Fundamentals

- Integrity
  - Integrity makes it difficult for a third party to substitute one message for another.
  - It allows the recipient of a message to verify it has not been modified in transit.
- Nonrepudiation
  - makes it difficult for the originator of a message to falsely deny later that they were the party that sent the message.
  - E.g., your signature on a document.
- Confidentiality
  - Traditionally, the primary objective of cryptography.

## Fundamentals

- Authentication is a service that
  - allows receivers of a messages to identify its origin
  - makes is difficult for third parties to masquerade as someone else
  - e.g., your driver's license and photo authenticates your image to a name, address, and birth date.
- Authorization is a service that
  - Allows only entities that have been authenticated and who appear on an access list to utilize a service.
  - E.g., your date of birth on your driver's license authorizes you to drink as someone who is over 21.

## Defns

- Cryptography is one tool (not the only) useful for providing Security services such as:
  - **Authorization**: conveyance of official sanction to do or be something to another entity.
  - **Access Control**: restricting access to resources to privileged entities.
  - **Availability**: ensuring a system is available to authorized entities when needed.
  - **Anonymity**: concealing the identity of an entity involved in some process
  - **Certification**: endorsement of information by a trusted entity.
  - **Revocation**: retraction of certification or authorization

## Fundamentals

- Access Control
  - ensures that specific entities may perform specific operations on a secure object.
  - In Unix, access control is specified with three operations:
    - Read
    - Write
    - Execute

    as applied to file or directory for an *owner*.
- Availability
  - ensures that a service or information is available to an (authorized) user upon demand and without delay.
  - *Denial-of-service attacks* seek to interrupt a service or make some information unavailable to legitimate users.

## Fundamentals

- Anonymity
  - Concealing the originator of a message within a set of possible entities.
    - The degree of anonymity of an entity is the sum chance that everyone else in the set is the originator of the message.
    - Anonymity is a technical means to privacy.
  - Privacy: concealing personal information, a form of confidentiality.
- Certification and Revocation
  - Just as important as certifying an entity, we need to be able to take those rights away, because of compromise, a change in policy, or the safety that comes from "refresh".

## Attacks

- Passive Attacks:
  - **Interception**: attacks confidentiality.
    - a.k.a., eavesdropping, "woman-in-the-middle" attacks.
  - **Traffic Analysis**: attacks confidentiality, or anonymity.
    - Can include traceback on a network, CRT radiation.
- Active Attacks
  - **Interruption**: attacks availability.
    - a.k.a., denial-of-service attacks
  - **Modification**: attacks integrity.
  - **Fabrication**: attacks authenticity.

## Administrators

- Persons managing the security of a valued resource consider
  - Prevention: taking measures that prevent damage.
    - E.g., one-time passwords (s/key)
  - Detection: measures that allow detection of when an asset has been damaged, altered, or copied.
    - E.g., intrusion detection, network forensics
  - Recovery: restoring systems that were compromised.
  - Risk assessment: the value of a resource should determine how much effort (or money) is spent protecting it.
    - E.g., If you have nothing in your house of value do you need to lock your doors other than to protect the house itself?
    - If you have an $16,000,000 artwork, you might consider a security guard. (can you trust the guard?)

## Secure or not?

- Steganography is "security by obscurity" and isn't secure
  - Steganography is the practice of hiding messages in other messages.
  - E.g., invisible ink, minute differences between handwritten characters.
  - http://www.outguess.org/, hide a message in a jpeg image.
- What does it mean for information to be secure?
  - The cost of breaking the security exceeds the value of the secured service or information.
  - The time required to break the security exceeds the useful lifetime of the information

## Cryptography

- The most widely used tool for securing information and services is cryptography.
- Cryptography relies on ciphers: *mathematical functions used for encryption and decryption of a message.*
  - *Encryption*: the process of disguising a message in such a way as to hide its substance.
  - Ciphertext: an encrypted message
  - Decryption: the process of returning an encrypted message back into plaintext.

## Ciphers

- The security of a cipher may rest in the secrecy of its *restricted* algorithm .
  - Whenever a users leaves a group, the algorithm must change.
  - Can't be scrutinized by people smarter than you.
  - But, secrecy is a popular approach :(
- Modern cryptography relies on *keys, a selected value from a large set (a keyspace), e.g., a 1024-bit number. $2^{1024}$ values!*
  - Security is based on secrecy of the key, not the details of the algorithm
  - Change of authorized participants requires only a change in key.

## Ciphers

For some message *M*, let's denote the encryption of that message into cipher text as

$$\{M\}_K = C$$

Similarly, the decryption into plain text as

$$\{C\}_K = M$$

Notice,

$\{\{M\}_K\}_K = M$ symmetric key algorithms (e.g. DES,AES).

Some algorithms use different keys for each operation:

$\{\{M\}_{K1}\}_{K2} = M$ public-key algorithms (e.g.,RSA).

## Public-key Crypto

- One problem with symmetric key algorithms is you need a secure method of telling your partner the key.
- Plus, you need a separate key for everyone you might communicate with.
- Public-Key algorithms use a public-key and private-key pair over a message.
  - Only the public-key can decrypt a message encrypted with the private key.
  - Similarly, only the private key can decrypt a message decrypted with the public key.

## Public-key Crypto

- If you know my public key, then only I can read your message.
- If you can decrypt a message with my public key, then I must have written it.
- First proposed by Whitfield Diffie and Martin Hellman, and independently by Ralph Merkle in the late 1970s.
- (And by the NSA some time before that.)
- Generally, these algorithms are much slower than symmetric key algorithms.

## Public-key Crypto

- A crucial feature is that the private key is difficult to determine from the public key, and the robustness against chosen-plaintext attacks.
- In practice, a hybrid of symmetric and public-key approaches is used.
- RSA is the most widely used public-key algorithm.

## Rivest, Shamir, and Adleman (RSA)

- Relies on the difficulty of factoring large numbers.
  - Choose two large prime numbers, *p* and *q*.
  - Compute n = pq and z = (p-1)(q-1).
  - Choose a number, e, less than n, which has no common factors (other than 1) with z. (e and z are relatively prime).
  - Find a number, d, such that *ed-1* is exactly divisible (i.e., with no remainder) by z

    $ed \equiv 1 \pmod{z}$.

  - The public key that made available to the world is the pair of numbers (n,e);
  - the private key is the pair of numbers (n,d).

## RSA

- Private key = (n,d)
- Public key = (n,e)

- Encryption of a character $M_i$

  $C_i = M_i^e \pmod{n}$
- Decryption of a cipher character $C_i$

  $M_i = C_i^d \pmod{n}$

## Example encryption and decrypt

- Choose two primes *p=47 and q=71.*
- Then, *n=pq= 3337.*
- Choose *e*, relatively prime to *z=46x70=3220*
  - how about 79.
- We need a *d* such that $ed \equiv 1 \pmod{z}$, in other words, $d = e^{-1} \bmod z$.
  - $d = 79^{-1} \pmod{3220} = 1019$
- Private key (3337,79). Private key (3337,1019)

- Encrypt 688 -> $688^{79} \bmod 3337 = 1570$.
- Decrypt 1570 -> $1570^{1019} \bmod 3337 = 688$

## Hashes

- Hashes are going to be a tool we use primarily for authentication.
- While related, these are not the same hashes you would use as the function in a hash table.
- They have stricter requirements.

## Method: Hash Functions

- A hash *H* is a *one-way* function that operates on arbitrary-length message *m*, and returns a fixed-length value *h*.

$$h=H(m)$$

- Given a message m, it is easy to compute H(m)
- Given *h*, it is hard to compute *m* such that *H(m)=h*.
- Given specific *m*, it is hard to find another message *m'*, such that *H(m)=H(m')*.
- Given a large set M, it's difficult to find any pair $(m_i,m_j)$ that hash to the same value.

- Hashes provide a fingerprint of *m*.

## Digital Signatures

- Real signatures provide a number of features
  - Signature provides authenticity for a documents
  - Signatures are "hard" to forge
  - Signatures, as parts of the document, aren't reusable.
  - Signatures are unalterable or erasable.
  - Signatures can't be repudiated.
- In reality there are ways around all of these for real signatures.

## Signing with Hash Functions

1. Alice produces a one-way has of the document.

   A: h=H(D)

2. Alice encrypts the hash

   A: $\{h\}K_{A-}$

3. Alice sends the document and the signed hash to Bob.

   A->B: D, $\{h\}K_{A-}$

4. Bob verifies by producing the same hash and decrypting the hash Alice sent.