CS342 Computer Security **Handout # 2**
Profs. Daniel Bilar and Lyn Turbak **Friday, Sep. 8, 2006**
Wellesley College

Symmetric-key Cryptography

Cast of Characters

- Alice wants to send a message M to Bob.
- Eve may be eavesdropping on the communication (passive attacker)
- Mallory may be able to change the message (active attacker)

Goals of Cryptography

1. *Confidentiality*: No one can read M except Alice and Bob (e.g., whispering, locked box);
2. *Authentication*: Bob knows M comes from Alice, not someone masquerading as Alice (e.g., wax seal, watermark paper, signature);
3. *Integrity*: Bob can verify that M has not be altered after it was sent by Alice (e.g., tamper-proof packaging);
4. *Nonrepudiation*: Alice should not be able to falsely deny sending M.

Terminology

Steganography = hiding messages, "security through obscurity". E.g., under wax, under hair, invisible ink, in lower order bits of images.

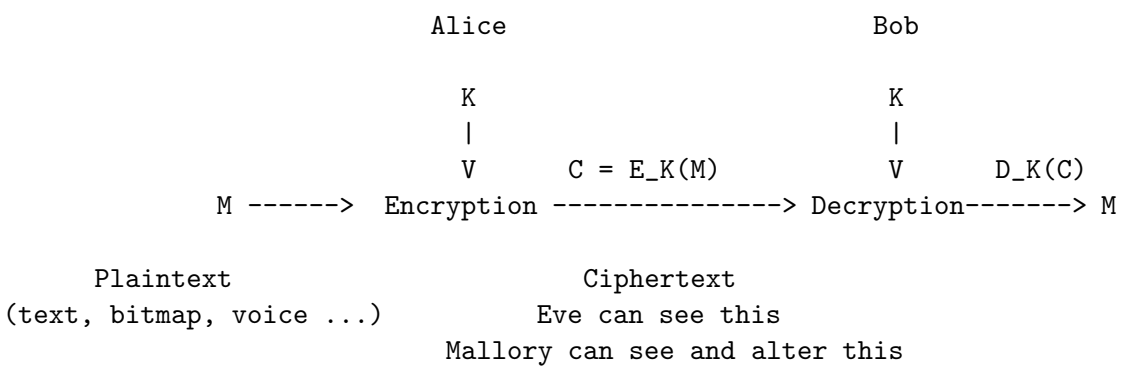
Cryptography = scrambling messages: converting messages into "gibberish" that can be converted back to message.

Cryptanalysis = breaking secret codes.

Cryptology = cryptography + cryptanalysis (in practice *cryptography* is often used for *cryptography*).

Note: Can combine steganography and cryptography.

The Basic Scenario for Cryptography



Kerckhoff's Principle

Separate cryptography into public algorithms and private keys (Dutch linguist Auguste Kerckhoff von Nieuwnenhof, 1883).

- Private algorithm is an example of "security through obscurity", which is usually not secure, at least once algorithm is known (e.g., Navajo code talkers in WW2).
- Proprietary algorithms often contain holes; public algorithms are analyzed by lots of smart people to find potential problems.
- People (rightly) suspicious of private algorithms, so hard to adopt on widespread basis.

Symmetric-Key vs. Public-Key Cryptography

- *Symmetric-key (secret-key)* (this lecture): Alice and Bob share the same key (or they have different keys that can be easily calculated from one another). Communicating keys is a big problem.
- *Asymmetric-key (public-key)* (Scout's lectures): Bob can publish a public key that anyone (including Alice) can use to encrypt a message, but only Bob has a private key that can decrypt the message. The private key cannot easily be determined from the public key. Alice can contact Bob without exchanging keys with him! But public-key management is still a problem ...

Simple Cryptography

For now, assume messages are letter sequences from 26-element alphabet.
In each example, what is the key? How many keys are there?

- *Shifting cipher*: e.g. decoder ring, Caesar cipher (shift left three characters)

plaintext	`alisaspy`
ciphertext	`dolvdvsb`

- *Substitution cipher* = bijective function on characters. E.g.

alphabet	`abcdefghijklmnopqrstuvwxyz`
permutation	`emqydwhlzaxtndjvucokfrpgib`
plaintext	`alisaspy`
ciphertext	`etzoeovi`

- *Transposition (permutation) cipher* = permute positions of message (anagram). E.g., "rail fence", Spartan scytale

plaintext	`alisaspy`
row1	`aiap`
row2	`lssy`
ciphertext	`aiaplssy`

Breaking Cryptography

- *Brute force attack* = try all possible keys. Feasability depends on key size, available computrons.
- More clever attacks use information theory. E.g., letter frequency information can break substitution cipher quickly. (To reduce redundancy of messages, some crypto implementations first compress message.)
- Attack classification:
  - *ciphertext-only*: are only given encrypted messages
  - *known-plaintext*: are given encrypted messages and associated plaintext
  - *chosen-plaintext*: can choose plaintext for encryption
  - *chosen-ciphertext*: deduce key from black-box decrytper
  - *purchase-key/rubber-hose*: bribe or threaten key holder until he gives up key.

Perfectly Secure Encryption: One-Time Pads

- A one-time pad is a random sequence of bits (effectively a really large key) shared by Alice and Bob. Messages are encrypted/decrypted by XORing with the pad:

```
      char:   |a       |l       |i       |s       |
      ascii:  |97      |108     |105     |115     |
      bits:   |01100001|01101100|01101001|01110011|
  ---------------------------------------------------
      pad:    |01111000|11110000|10110000|00011000|
  ---------------------------------------------------
ciphertext:   |00011001|10011100|11011001|01101011|
```

- *stream ciphers* are based on this idea – key is used to seed a psuedorandom number generator, and the generated bits are used like a one-time pad. E.g., RC4 stream cipher used in SSL (basis of web-browser security)
- Things can go wrong if:
  - pad is reused (see PS1)
  - pad isn't truly random. If only pseudorandom, then all information is in the seed (e.g. early Netscape SSL stream cipher seed easy to guess)

Data Encryption Standard (DES)

- In 1972, National Bureau of Standards issued request for standard crypto algorithm.
- Data Encryption Standard (DES) was adopted as federal standard in 1977 and ANSI standard in 1981. Grew out of IBM Lucifer system and evaluated by the National Security Agency (NSA). People worried that NSA reduced key size from 128 to 56 bits and may have introduced a trapdoor.
- DES is a block cipher — maps 64-bit plaintext block to 64 bit ciphertext block controlled by 56-bit key. Uses 16 rounds of the same substitution/permutation operation.
- People suspected DES was breakable by brute force with enough computing power. This was shown in series of RSA Labs sponsored challenges to break DES keys:
  - Jun. 1997: 140 days by distributed.net
  - Feb. 1998: 39 days by distributed.net
  - Jul. 1998: 56 hours by Electronic Frontier Foundation's (EFF) "Deep Crack" machine
  - Jan. 1999, key broken in 22 hours and 15 minutes by Deep Crack + distributed.net

  Key size matters!

Other Block Ciphers

- National Institute of Standards and Technology (NIST) requested proposals for new encryption standard in 1997; winner in 2000 was dubbed Advanced Encryption Standard (AES). Minimum of 128-bit keys; up to 256-bit keys.
- Other block ciphers:
  - Skipjack (Clipper chip/Fortezza program, 80-bit keys)
  - Triple-DES = three rounds of DES with 3 different keys; 2*56 = 112 effecttive bits of security with 3*56 = 168-bit keys.
  - IDEA (128-bit keys)
  - Blowfish (Schneier, up to 448-bit keys).
  - RC5 (Rivest, variable key length)

Encryption/Decryption Examples using Linux's `openssl` Command

```
[cs342@puma] cat secret.txt
One if by land, two if by sea.

[cs342@puma] openssl enc -des -nosalt -pass pass:foobar -in secret.txt -out secret.enc

[cs342@puma] cat secret.enc
S\3725\337*|^T\341\345^^\366\316\207\370\261\351d\371  ^D^A#\245^V1>\240Gy\230\256

[cs342@puma] openssl enc -d -des -nosalt -pass pass:foobar -in secret.enc
One if by land, two if by sea.
```

Message Authentication

Want a way to determine that an unencrypted message is authentic – i.e., it is really from who it says it's from, and has not been changed.

One-Way Hash Functions

- A *hash function* $H(M)$ returns a fixed-length hash value h for any size of message M.
  - E.g., for producing keys in a hash table.
  - Because h is generally (much) smaller than M, there will generally be (very) many messsages with the same hash value h.
- A hash function $H(M)$ is *one-way* if:
  - it's easy to computer $H(M) = h$.
  - given h, it's hard to find an M' such that $H(M') = h$. (This implies that given M, it's hard to find M' such that $H(M') = H(M)$.)

  h is known as a *cryptographic checksum*, *fingerprint*, *message digest*. Changing a single bit in M changes half the bits in h on average.
- Examples:
  - MD5 gives a 128 bit fingerprint
  - SHA1 gives a 160 bit fingerprint

Hashing Examples using Linux's `openssl` Command

```
[cs342@puma] echo "One if by land, two if by sea." | openssl dgst -md5
fe66cbf9d929934b09cc7e8be890522e

[cs342@puma] echo "One if by land, two if by sea." | openssl dgst -md5 -c
fe:66:cb:f9:d9:29:93:4b:09:cc:7e:8b:e8:90:52:2e

[cs342@puma] echo "One if by land, two if by sea" | openssl dgst -md5 -c
78:4b:61:4a:66:98:17:82:18:d9:25:ca:c9:64:c5:56

[cs342@puma] echo "One if by land, Two if by sea." | openssl dgst -md5 -c
96:07:e8:69:cd:97:59:98:ad:21:8e:46:a8:c0:4f:0e

[cs342@puma] echo "One if by land, two if by sea." | openssl dgst -sha1 -c
28:47:d1:e5:9a:96:83:bf:2f:2a:91:b8:f3:ec:21:63:d3:be:5a:6b

[cs342@puma] echo "One if by land, two if by sea" | openssl dgst -sha1 -c
24:e2:d1:19:44:c2:17:49:1f:d8:9c:23:d0:9d:d2:d9:87:87:11:f1
```

Message Authentication Code (MAC)

- One-way hashes can be used for integrity in some cases (e.g., code fingerprint from "reputable" website), but they're not suitable for Alice and Bob. Why?
- A MAC combine hashes with keys to so that only key-holders can authenticate. Useful for authenticating files between users and determining if user files have changed
- Basic idea: send pair $\langle M, S \rangle$ of message M and signature S, where the S calculated from M and key K. Some possible signatures:
  - encrypt hash value of M with key: $S = E_K(H(M))$.
  - hash encrypted value of M: $S = H(E_K(M))$.
  - hash result of concatenating key and message: (1) $S = H(K, M)$ or (2) $S = H(M, K)$. Problem with (1) is that Mallory can add new blocks to end of message.

  Schneier mentions some vulnerabilties of these approaches.

Collision Resistance Thwarts the Birthday Attack

- A hash function is *collision resistant* if it's hard to find two random messages M and M' such that $H(M) = H(M')$.
- Collision resistance is important to foil *birthday attack*.