CSE 484 (Winter 2008)

Applied Cryptography

Tadayoshi Kohno

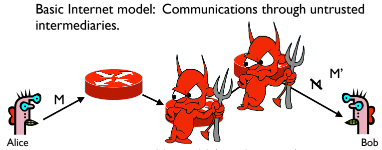
Thanks to Dan Boneh, Dieter Gollmann, John Manferdelli, John Mitchell, Vitaly Shmatikov, Bennet Yee, and many others for sample slides and materials ...

Goals for Today

- Cryptography Background
- Symmetric (Shared-Key Foundations)

Basic Problem

Basic Internet model: Communications through untrusted intermediaries.



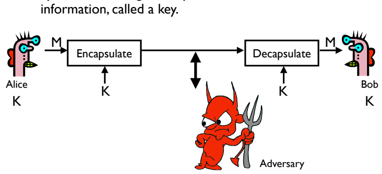
I know M (attack privacy)
I can change M (attack integrity)

Important for: Secure remote logins, file transfers, web access, ...

Symmetric Setting

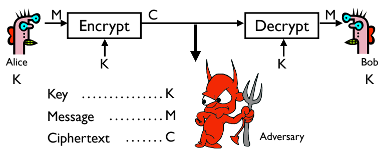
Solution: Encapsulate and decapsulate messages in some secure way.

Symmetric setting: Both parties share some secret information, called a key.



Achieving Privacy

Encryption schemes



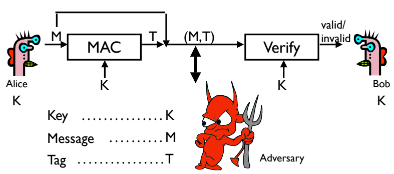
History

- Substitution Ciphers
  - Caesar Cipher
- Transposition Ciphers
- Codebooks
- Machines

- Recommended Reading: The Codebreakers by David Kahn.
  - Military uses
  - Rumrunners
  - ....

Achieving Integrity

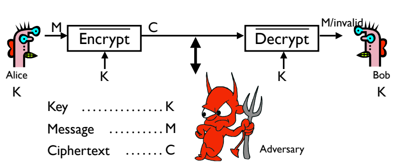
Message authentication schemes or message authentication codes or MACs



Achieving Both Privacy and Integrity

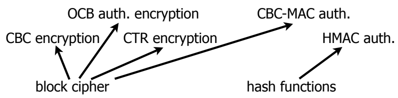
Authenticated encryption scheme

(Authenticated encryption notion is "new" (around 2000), so many books and protocols don't discuss this. Can be subtle!!!)



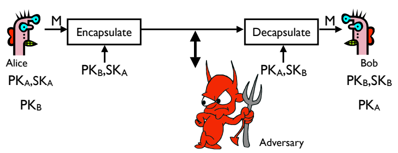
How this is achieved

- Layered approach:
  - Cryptographic primitives, like block ciphers, stream ciphers, and hash functions
  - Cryptographic protocols, like CBC mode encryption, CTR mode encryption, HMAC message authentication
- Today:
  - Start on the above. Basic concepts. Basic pitfalls.

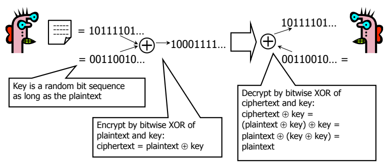


Asymmetric Setting (NOT today)

Asymmetric setting: <u>Public</u> and <u>Secret</u> keys. (Can help establish shared secret keys K.)



One-Time Pad

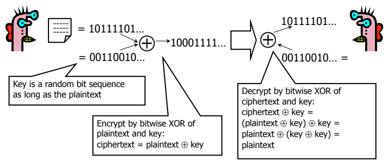


Cipher achieves perfect secrecy if and only if
there are as many possible keys as possible plaintexts, and
every key is equally likely (Claude Shannon)

Advantages of One-Time Pad

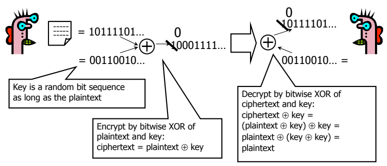
- Easy to compute
  - Encryption and decryption are the same operation
  - Bitwise XOR is very cheap to compute
- As secure as theoretically possible
  - Given a ciphertext, all plaintexts are equally likely, regardless of attacker's computational resources
  - …as long as the key sequence is truly random
    - True randomness is expensive to obtain in large quantities
  - …as long as each key is same length as plaintext
    - But how does the sender communicate the key to receiver?

Disadvantages



Disadvantage #1: Keys as long as messages.
Impractical in most scenarios
Still used by intelligence communities

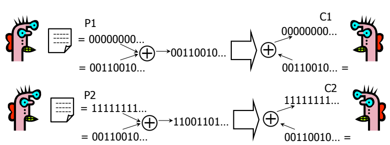
Disadvantages



Disadvantage #2: No integrity protection

Disadvantages

Disadvantage #3: Keys cannot be reused

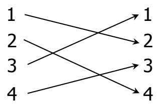


Learn relationship between plaintexts:
$C1 \oplus C2 = (P1 \oplus K) \oplus (P2 \oplus K) = (P1 \oplus P2) \oplus (K \oplus K) = P1 \oplus P2$

Reducing Keysize

- What do we do when we can't pre-share huge keys?
  - When OTP is unrealistic
- We use special cryptographic primitives
  - Single key can be reused (with some restrictions)
  - But no longer provable secure (in the sense of the OTP)
- Examples: Block ciphers, stream ciphers

Background: Permutation

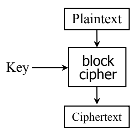


CODE becomes DCEO

- For N-bit input, N! possible permutations
- Idea: split plaintext into blocks, for each block use secret key to pick a permutation, rinse and repeat
  - Without the key, permutation should "look random"

Block Ciphers

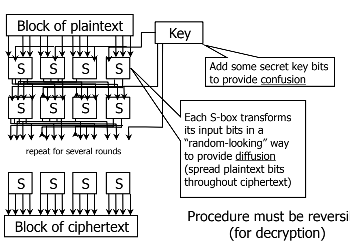
- Operates on a single chunk ("block") of plaintext
  - For example, 64 bits for DES, 128 bits for AES
  - Same key is reused for each block (can use short keys)



Block Cipher Security

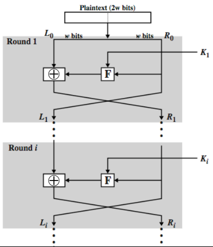
- Result should look like a random permutation
  - "As if" plaintext bits were randomly shuffled
- Only computational guarantee of secrecy
  - Not impossible to break, just very expensive
    - If there is no efficient algorithm (unproven assumption!), then can only break by brute-force, try-every-possible-key search
  - Time and cost of breaking the cipher exceed the value and/or useful lifetime of protected information

Block Cipher Operation (Simplified)



Procedure must be reversible (for decryption)

Feistel Structure (Stallings Fig 2.2)



DES

- Feistel structure
  - "Ladder" structure: split input in half, put one half through the round and XOR with the other half
  - After 3 random rounds, ciphertext indistinguishable from a random permutation (Luby & Rackoff)
- DES: Data Encryption Standard
  - Feistel structure
  - Invented by IBM, issued as federal standard in 1977
  - 64-bit blocks, 56-bit key + 8 bits for parity

DES and 56 bit keys (Stallings Tab 2.2)

- 56 bit keys are quite short

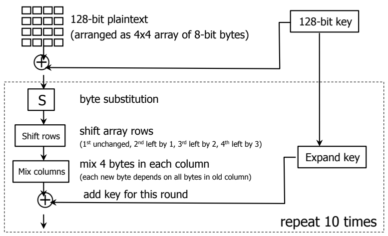
Key Size (bits)	Number of Alternative Keys	Time required at 1 encryption/μs	Time required at 10^6 encryptions/μs
32	$2^{32} = 4.3 \times 10^9$	$2^{31}\ \mu s = 35.8$ minutes	2.15 milliseconds
56	$2^{56} = 7.2 \times 10^{16}$	$2^{55}\ \mu s = 1142$ years	10.01 hours
128	$2^{128} = 3.4 \times 10^{38}$	$2^{127}\ \mu s = 5.4 \times 10^{24}$ years	5.4×10^{18} years
168	$2^{168} = 3.7 \times 10^{50}$	$2^{167}\ \mu s = 5.9 \times 10^{36}$ years	5.9×10^{30} years
26 characters (permutation)	$26! = 4 \times 10^{26}$	$2 \times 10^{26}\ \mu s = 6.4 \times 10^{12}$ years	6.4×10^6 years

- 1999: EFF DES Crack + distibuted machines
  - < 24 hours to find DES key
- DES ---> 3DES
  - 3DES: DES + inverse DES + DES (with 2 or 3 diff keys)

Advanced Encryption Standard (AES)

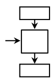
- New federal standard as of 2001
- Based on the Rijndael algorithm
- 128-bit blocks, keys can be 128, 192 or 256 bits
- Unlike DES, does <u>not</u> use Feistel structure
  - The entire block is processed during each round
- Design uses some very nice mathematics

Basic Structure of Rijndael



Encrypting a Large Message

- So, we've got a good block cipher, but our plaintext is larger than 128-bit block size
- Electronic Code Book (ECB) mode
  - Split plaintext into blocks, encrypt each one separately using the block cipher



- Cipher Block Chaining (CBC) mode
  - Split plaintext into blocks, XOR each block with the result of encrypting previous blocks
- Counter (CTR) mode
  - Use block cipher to generate keystream, like a stream cipher
- …