SE 4472a

Information Security

Week 2:
Security Goals and Adversarial Models

Fall 2014
Prof. Aleksander Essex

Western Engineering

Some keywords

- Plaintext
  - The message that is being encrypted
- Ciphertext
  - The encrypted message
- Key
  - The secret that is used to transform the plaintext into the ciphertext
- Keyspace
  - The set of possible keys

Computational vs. Perfect secrecy

The one-time pad

Computational Secrecy

- Brute-force search: crack the code by trying all possible keys
- Fundamental approach: an adversary can always perform this attack
- Goals of cryptosystem designers:
  1. **Key space exponential in key length.** Make number of possible keys grow exponentially in the length of the key. Making the key one bit longer should double the number of possible keys an attacker has to try.
  2. **Infeasible to brute force.** Make keyspace too big for any real-world adversary to feasibly brute force. In concrete, contemporary terms, we're talking $>2^{100}$
  3. **Brute force=worst case.** Design the cryptosystem such that brute-force is the best strategy for the adversary, i.e., no shortcuts due to weaknesses

Computational Secrecy

- Even if Enigma's design and operation wasn't flawed, you could still just try all of the rotor/plug combinations
- 3 rotors in one of 26 positions: 26^3 possible keys
  - With everything else (plugboard, etc) about 2^{70}
- This is *computable*, even if not quite feasible to brute force
- Is there a stronger model?
  - i.e., something that even infinite computational power can't crack?

The one-time pad

- Era: Early to mid 20th century
- Perfect secrecy (Claude Shannon)

Message	`whatcouldpossiblygowrong`	
Pad	`kjpvmqhfuzpbmwmxinlbfjok`	+
Ciphertext	`gqpooebqxodteenigtzxwxbq`	

The one-time pad

- Claim: If the following three properties hold, the one-time pad is perfectly hiding:
  1. Pad is chosen independently at random
     1. Every possible pad is chosen with equal likelihood
  2. Pad is exactly as long as the message
  3. Pad is only ever used once
     1. Note: we mean every time a message is encrypted, a fresh pad is generated independently at random. It is still statistically possible to generate the same pad twice, though no more likely than drawing any pair of possible pads.
     2. Pad can never be leaked and must be destroyed after use

Perfect Secrecy

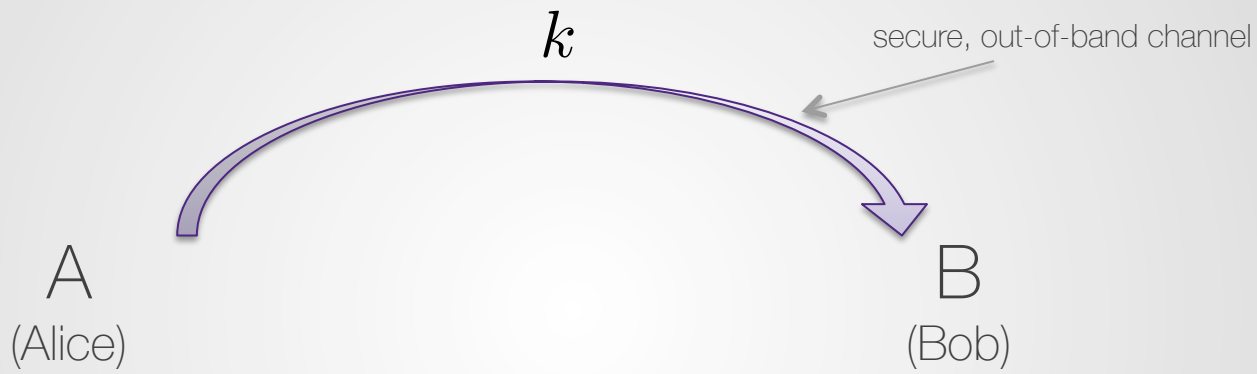
- One-time pad is perfectly (*information-theoretically*) hiding
  - No amount of computers can help you since ciphertext can decrypt to any message given corresponding pad (and all pads are equally likely)
- If it's so secure, why doesn't everyone use it?
  - Major practical issues
    - Generating, transporting and storing pad
    - |pad| = |message|
    - Human error (pad reuse, pad alignment, etc)

Secrecy in this course

- Computational secrecy
  - Short fixed-length key
  - Crackable with enough computing power
  - Too many potential keys feasibly search
- Perfect (information-theoretic secrecy)
  - Long, message-length key
  - Cannot be cracked, even with infinite computing power since all messages are equally possible
  - Key transfer and management totally impractical for most real-world applications
- From this point on we will only consider *computationally hiding* cryptosystems.

Adversarial models

Model of communication



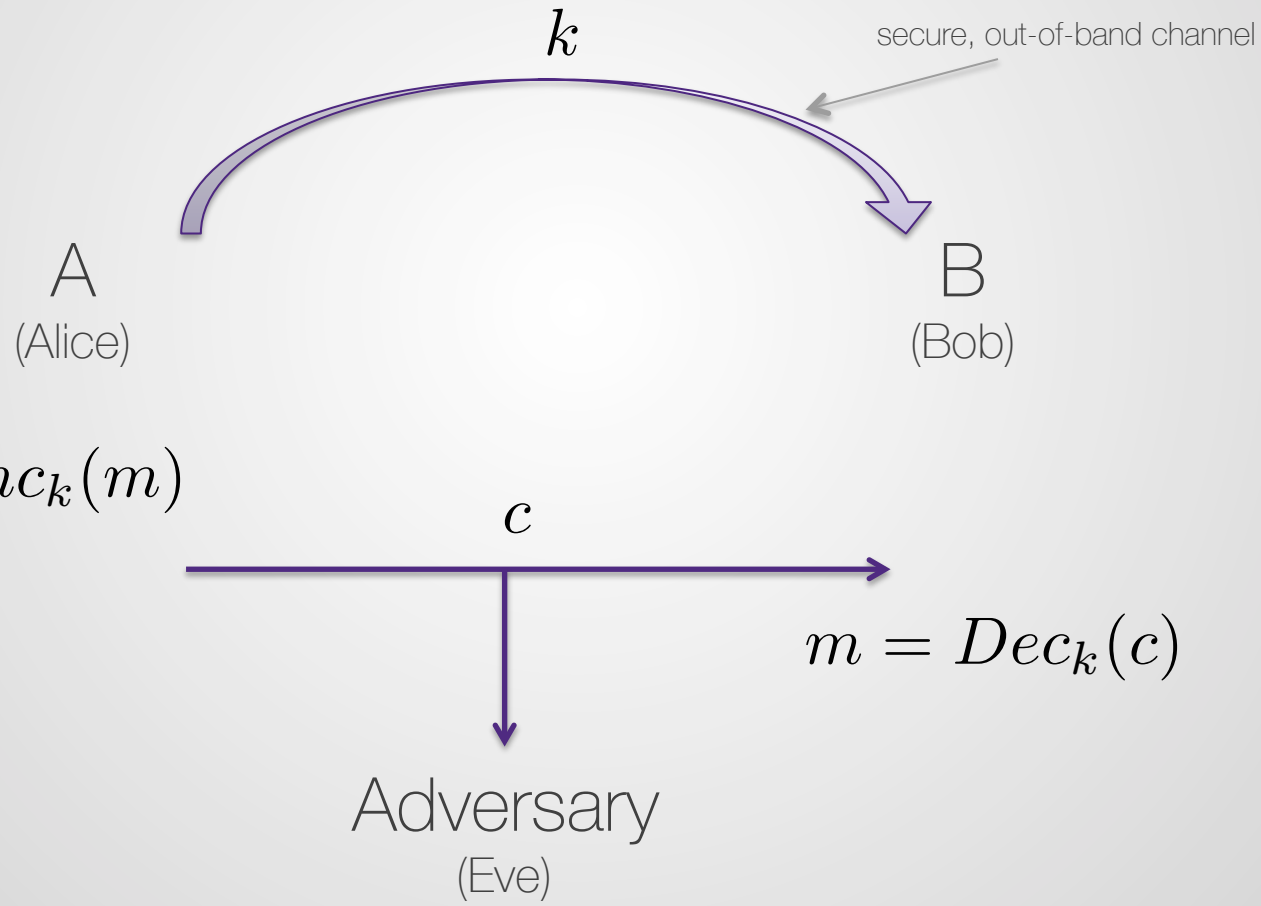
$c = Enc_k(m)$

c

$m = Dec_k(c)$

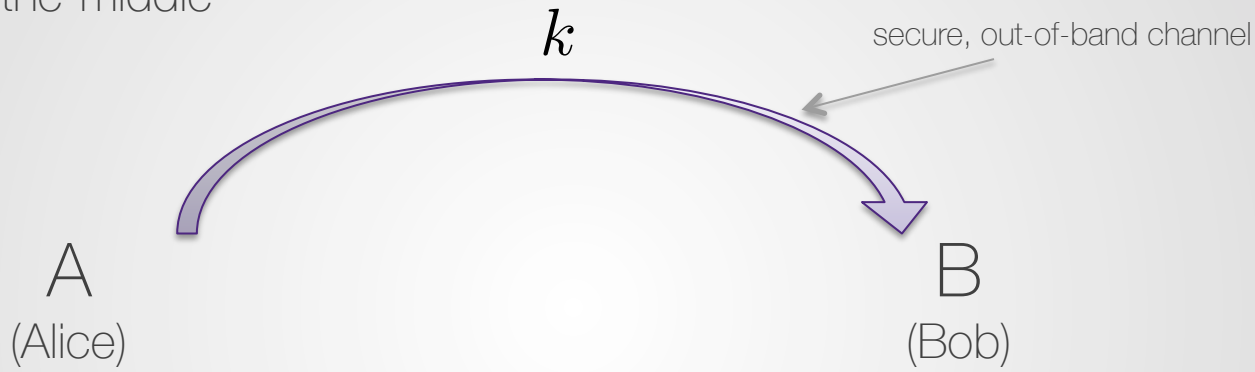
Passive adversary

- Adversary passively observes ciphertexts



Active (malicious) adversary

- Can modify or replace ciphertexts
- "Man-in-the-middle"



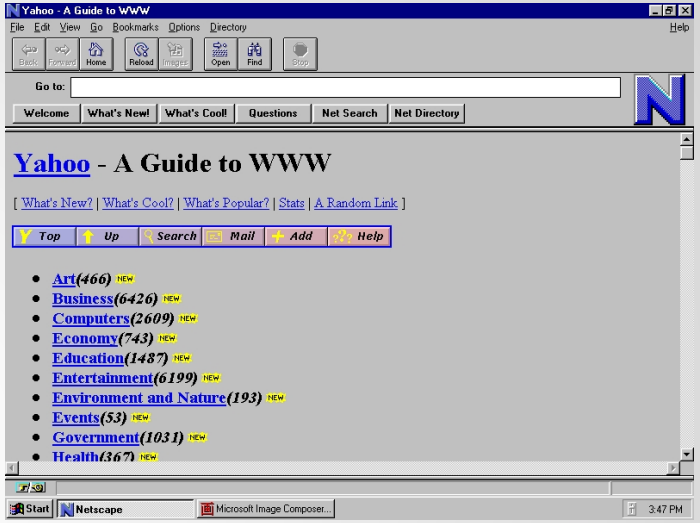
$c = Enc_k(m)$ $\xrightarrow{c}$ Adversary $\xrightarrow{c'}$ $m' = Dec_k(c')$

(Eve)

Security Goals

Scenario: It's the early 1990s

- The web just happened and it looks like this →
- People are starting to say "hey, we should sell stuff *online*"
- There's basically no online security yet



http://www.w3.org/2010/Talks/0921-html5-plh/web10.html

Discussion: What are the security problems that need to be solved? What do we *need* to make e-commerce happen?

Security Goals

In this course we will focus on the “CIA” triad of security goals:

- Confidentiality
  - Keep secret things secret
- Integrity
  - Keep secret things from being undetectably modified
- Authentication
  - Source authentication: know who you’re talking to
  - Message authentication: know who the message came from

All-for-one, one-for-all

- Can you have one property without another?
- Example: can you have confidentiality without integrity?
- Scenario: Alice encrypts a message. On the way over to Bob the ciphertext is modified by Eve.

All-for-one, one-for-all

- At first glance it seems like all Eve has managed to do is corrupt the message for Bob. But if Bob replies to Alice “hey, this message is corrupt” then Eve learns something.
- She can try again until Bob doesn’t respond with an error, which will give her information about what the plaintext must have been. We will study this later: it’s called the *padding oracle* attack.

All-for-one, one-for-all

- As we will see, we need confidentiality, integrity and authenticity *at the same time*.
- We cannot guarantee one property without the others
- We will study mechanisms for each:
  - **Confidentiality:** encryption, public-key exchange
  - **Integrity:** message authentication codes, hashes
  - **Authenticity:** digital signatures, certificates, public key infrastructure