Last time

- Security in Networks
  - Network Security Controls
  - Firewalls
  - Honeypots
  - Intrusion Detection Systems

This time

[Stinson, Shmatikov-Boneh]

- Internet Application Security and Privacy
  - Basics of cryptography
  - Symmetric-key encryption

Cryptography

- What is cryptography?
- Related fields:
  - Cryptography ("secret writing"): Making secret messages
    - Turning plaintext (an ordinary readable message) into ciphertext (secret messages that are "hard" to read)
    - The point of cryptography is to send secure messages over an insecure medium (like the Internet).
  - Cryptanalysis: Breaking secret messages
    - Recovering the plaintext from the ciphertext
- Cryptology is the science which studies these both

The scope of these lectures

- The goal of the cryptography unit in this course is to show you what cryptographic tools exist, and information about *using these tools in a secure manner*

- *We won't be showing you details of how the tools work.*

Some names to remember

When talking about cryptography, we often use a standard cast of characters

- Alice, Bob, Carol, Dave
  - People (usually honest) who wish to communicate
- Eve
  - A passive eavesdropper, who can listen to any transmitted messages
- Mallory
  - An active Man-In-The-Middle, who can listen to, and modify, insert, or delete, transmitted messages
- Trent
  - A Trusted Third Party

Building blocks

- Cryptography contains three major types of components
  - Secrecy components
    - Preventing Eve from reading Alice's messages
  - Integrity components
    - Preventing Mallory from modifying Alice's messages
  - Authenticity components
    - Preventing Mallory from impersonating Alice

Kerckhoffs' Principle (19th c.)

The security of a cryptosystem should not rely on a secret that's hard (or expensive) to change.

- So don't have secret encryption methods.
  - Then what do we do?
  - Have a large class of encryption methods, instead.
    - Hopefully, they're all equally strong.
  - Make the class public information
  - Use a secret key to specify which one you're using
  - It's easy to change the key; it's usually just a smallish number.

Kerckhoffs' Principle (19th c.)

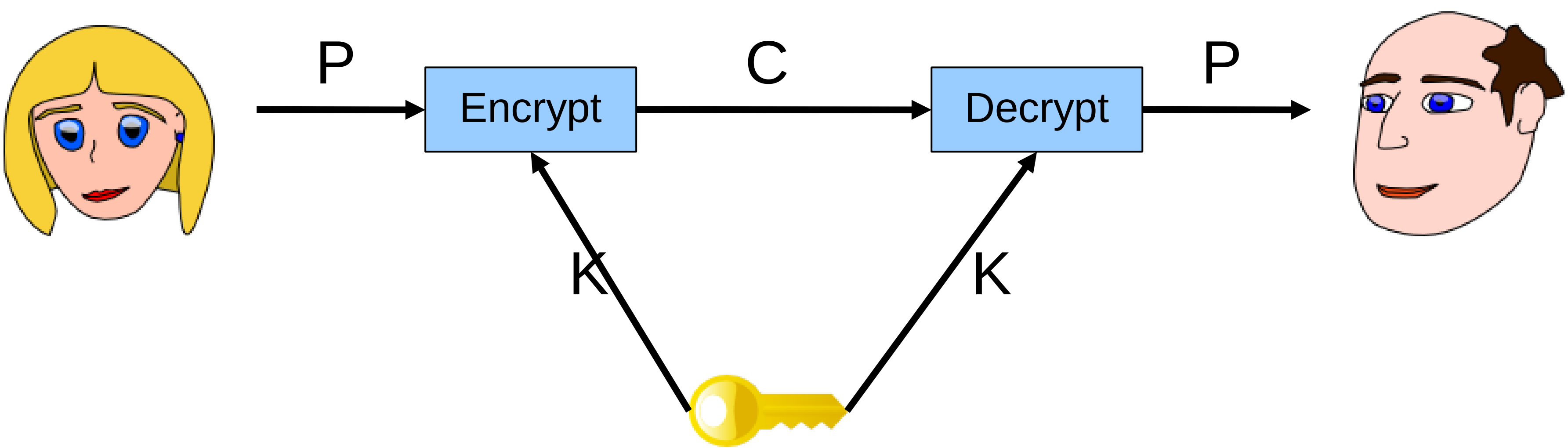
- This has a number of implications:
  - The system is at most as secure as the number of keys
  - Eve can just try them all, until she finds the right one
  - A strong cryptosystem is one where that's the best Eve can do
    - With weaker systems, there are shortcuts to finding the key
  - Example: some cryptosystems have 2^{120} possible keys
  - But you don't try them all; it's way easier than that!

Strong cryptosystems

- What information do we assume the attacker (Eve) has when she's trying to break our system?
- She may:
  - Know the algorithm (the public class of encryption methods)
  - Know some part of the plaintext
  - Know a number (maybe a large number) of corresponding plaintext/ciphertext pairs
  - Have access to an encryption and/or decryption *oracle*
- And we still want to prevent Eve from learning the key!

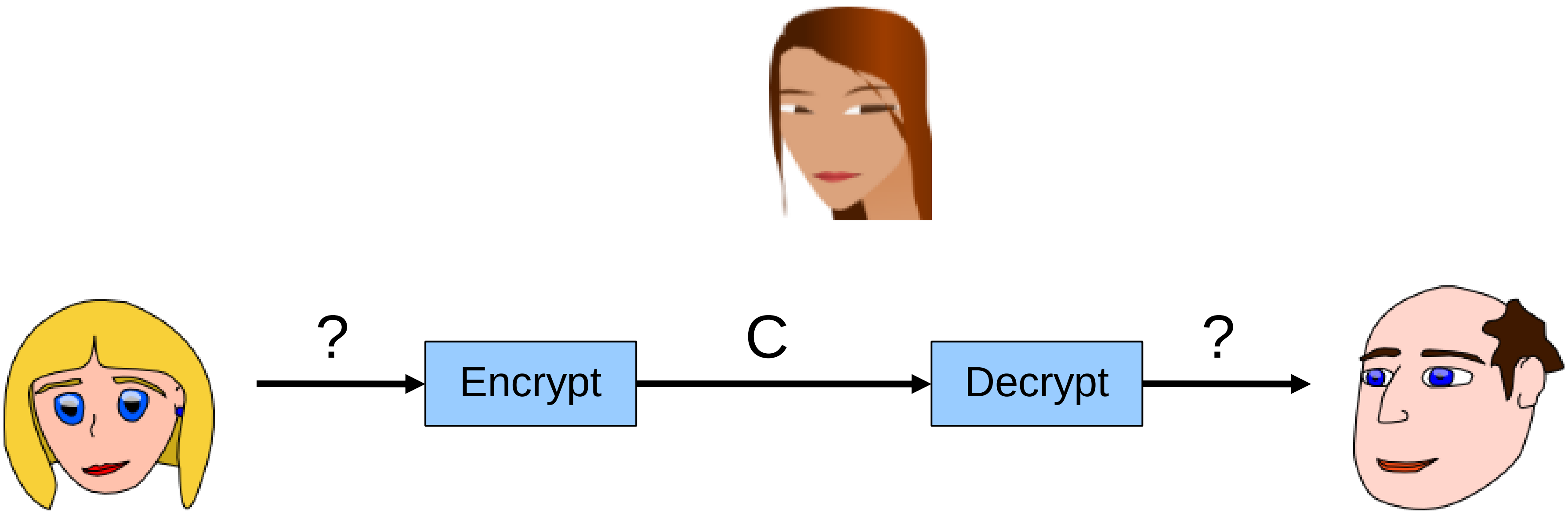
Symmetric encryption

- Symmetric encryption is the simplest form of cryptography
- Used for thousands of years
- The key Alice uses to encrypt the message is the same as the key Bob uses to decrypt it



Symmetric encryption

- Eve, not knowing the key, should not be able to recover the plaintext



Perfect symmetric encryption

- Is it possible to make a completely unbreakable cryptosystem?
- Yes: the One-Time Pad
- It's also very simple:
  - The key is a truly random bitstring of the same length as the message
  - The “Encrypt” and “Decrypt” functions are each just XOR

One-time pad

- But! It's very hard to use correctly
  - The key must be truly random, not pseudorandom
  - The key must never be used more than once!
    - A “two-time pad” is insecure!
- Used in the Washington / Moscow hotline for many years
- Q: Why does “try every key” not work here?
- Q: How do you share that much secret key?

Computational security

- In contrast to OTP's “perfect” or “information-theoretic” security, most cryptosystems have “computational” security
  - This means that it's certain they can be broken, given enough work by Eve
- How much is “enough”?
- At worst, Eve tries every key
  - How long that takes depends on how long the keys are
  - But it only takes this long if there are no “shortcuts”!

Some data points

- Assume that one computer can try about 17 million keys per second.
- A medium-sized corporate or research lab may have 100 computers.
- The Defense Advanced Research Projects Agency (DARPA) has 2 million computers.

- Remember that most computers are idle most of the time (they're waiting for you to type something); getting them to crack keys in their spare time doesn't actually cost anything extra.

40-bit crypto

This was the US legal export limit for a long time

2^{40} = 1,099,511,627,776 possible keys

- One computer: 18 hours
- One lab: 11 minutes
- BOINC: 30 ms

56-bit crypto

This was the US government standard (DES) for a long time

2^{56} = 72,057,594,037,927,936 possible keys

- One computer: 134 years
- One lab: 16 months
- BOINC: 36 minutes

128-bit crypto

This is the modern standard

2^{128} =

340,282,366,920,938,463,463,374,607,431,768,211,456 possible keys

- One computer: 635 thousand million million million years
- One lab: 6 thousand million million million years
- DARPA: 300 thousand million million years

Well, we cheated a bit

- This isn't really true, since computers get faster over time
  - A better strategy for breaking 128-bit crypto is just to wait until computers get 2^{88} times faster, then break it on one computer in 18 hours.
  - How long do we wait? Moore's law says 132 years.
  - If we believe Moore's law will keep on working, we'll be able to break 128-bit crypto in 132 years (and 18 hours) :-)
    - Q: Do we believe this?

An even better strategy

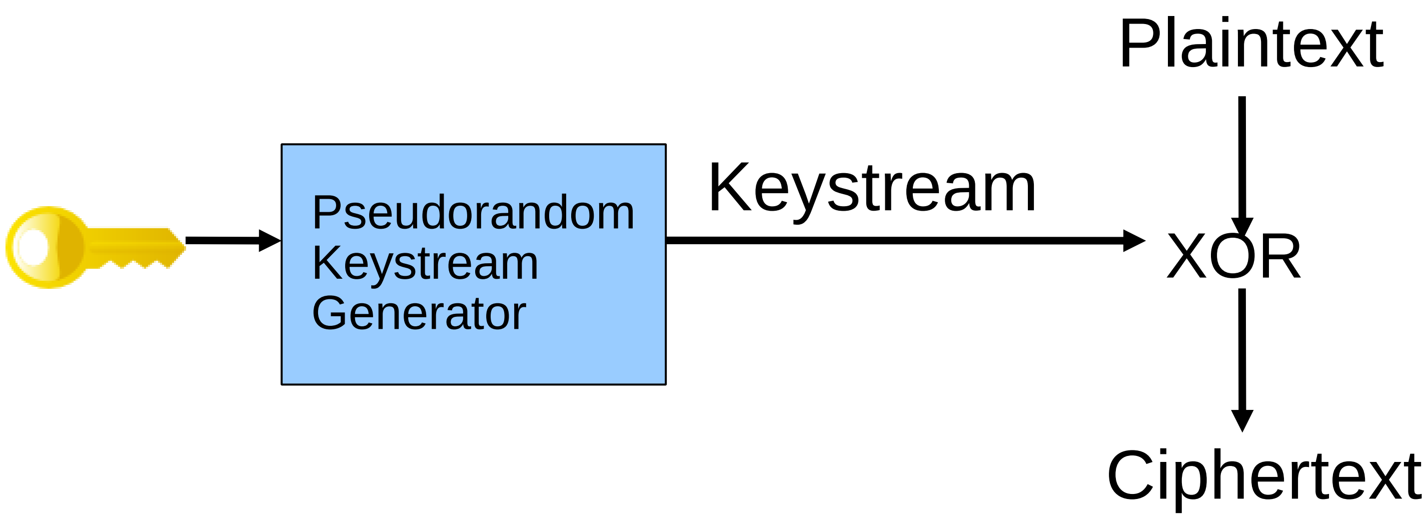
- Don't break the crypto at all!
- There are always weaker parts of the system to attack
    - Remember the Principle of Easiest Penetration
- The point of cryptography is to make sure the information transfer is not the weakest link

Types of symmetric ciphers

- Symmetric ciphers come in two major classes
  - Stream ciphers
  - Block ciphers

Stream ciphers

- A stream cipher is what you get if you take the One-Time Pad, but use a pseudorandom keystream instead of a truly random one



- RC4 is the most commonly used stream cipher on the Internet today

Stream ciphers

- Stream ciphers can be very fast
  - This is useful if you need to send a lot of data securely
- But they can be tricky to use correctly!
  - What happens if you use the same key to encrypt two different messages?
- WEP, PPTP are great examples of how not to use stream ciphers

Block ciphers

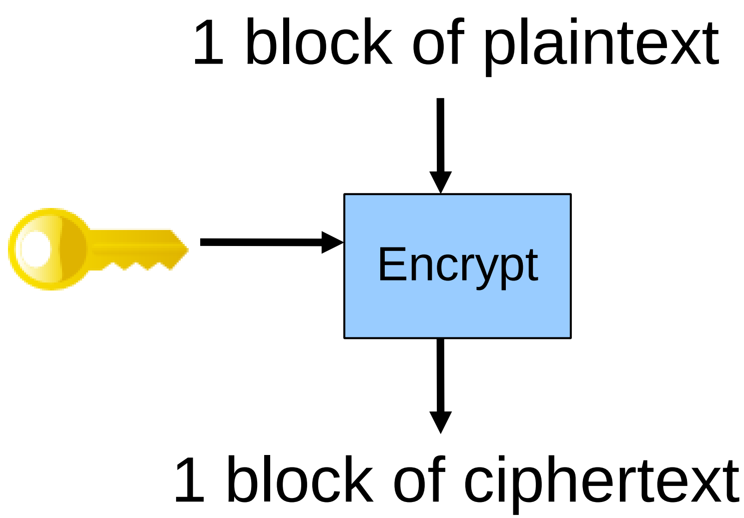
- Notice what happens in a stream cipher if you change just one bit of the plaintext
  - This is because stream ciphers operate on the message one bit at a time

- We can also use block ciphers
  - Block ciphers operate on the message one block at a time
  - Blocks are usually 64 or 128 bits long

- AES is the block cipher that is used today

Modes of operation

- Block ciphers work like this:



- But what happens when the plaintext is larger than one block?
    - The choice of what to do with multiple blocks is called the mode of operation of the block cipher

Modes of operation

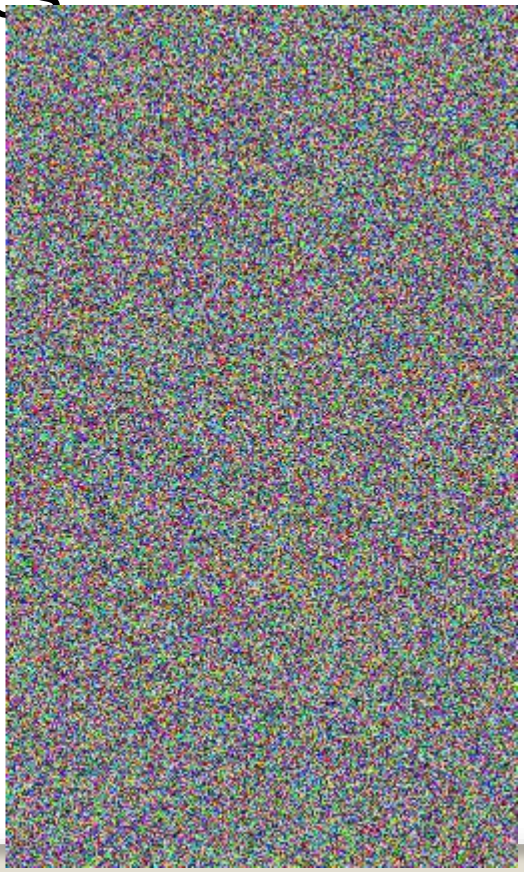
- The simplest thing to do is just to encrypt each successive block separately.
  - This is called Electronic Code Book (ECB) mode
- But if there are repeated blocks in the plaintext, you'll see the same repeating patterns in the ciphertext:

Modes of operation

- There are much better modes of operation to choose from
  - Common ones include Cipher Block Chaining (CBC) and Counter (CTR) modes

- Patterns in the plaintext are no longer exposed

- But you need an IV (Initial Value), which acts much like a salt



Recap

- Internet Application Security and Privacy
  - Basics of cryptography
  - Symmetric-key encryption

Next time

- Internet Application Security and Privacy
  - Public-key encryption
  - Integrity
  - Authentication