Last time

Classical cryptography

Cryptanalysis

- Kerckhoffs's principles
- attack models
- exhaustive key search
- frequency analysis

Frequency analysis

Example of a ciphertext-only attack

- don't know the plaintext
- know something *about* the plaintext: symbol frequencies
- English: 12% e, 9% t, etc., common groupings: the, an, qu...

Exercise: decode shift-enciphered ciphertext

hijstcih hdbtixbth tcrdst pcs strdst bthhpvth

Today

One-time pad

Block ciphers and modes

Vigenère cipher

State of the art from 17th–19th C

- defeated frequency analysis through *polyalphabetic* key
- longer key (e.g., a word) meant larger key search space

Example: "good night vienna" + "secrets" → "ysqu rbyzx xzigfs"

The fall of the Vigenère

Doesn't *actually* defeat frequency analysis

- can analyze frequency of every n^{th} letter
- can vary value of n
- doesn't stand up to automation

Exercise: "jlijzrjbz sdfavqs tt jbjqsyx kjzosu"* (hint: $n = 3$)

* or "jlijzrjbz jlgrdrj cl jbjqsyx skqwtl", depending on how you treat spaces...

The Vigenère rides again?

What if n was too large to admit frequency analysis?

What if the key was as long as the plaintext?

- one ciphertext symbol per key symbol ⇒ no frequency analysis
- *one-time pad* perfectly secure **iff** key symbols truly unpredictable
- true randomness is hard; **distributing** large keys is hard
- some utility in the real world; inspiration for *stream ciphers*

You will *not* hear me use the word "perfectly" very often in this course! However, in this case, it can be mathematically shown to apply _____.

Cipher security principles

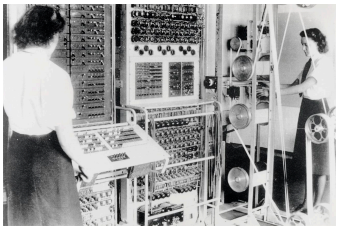
1. Keys must be **large** and **randomly generated**
2. Ciphertext has no mathematical or statistical relationship with plaintext **or** key
3. Best attack should be exhaustive key search
4. Kerckhoffs' Principle:
   cipher security must not depend on algorithmic secrecy

WWII: cryptographic arms race

Enigma

Turing

Bombes



Colossus

The Enigma machine was a substitution cipher, but *polyalphabetic*. It started with a relatively small keyspace, allowing _____ by a staff of Polish cryptanalysts. Over time, however, security improvements were made, greatly expanding the keyspace of the machine:

Version	Possible setups / keys
1920s	$26^3 \times 3! = 105.5 \times 10^3$
1930	$26^3 \times 3! \times \binom{26}{6} = 100.4 \times 10^9$
1939	$26^3 \times \binom{5}{3} \times 3! \times \binom{26}{6} = 1.5 \times 10^{19}$
1939 (navy)	$26^3 \times \binom{8}{3} \times 3! \times \binom{26}{10} = 8.4 \times 10^{19}$
1941 (navy)	$26^3 \times \binom{8}{3} \times 3! \times \binom{26}{10} = 1.8 \times 10^{20}4$
1942 (navy)	fourth optional rotor... larger keyspace

These later versions had keyspaces larger than 2^{66}... that's larger than the Data Encryption Standard used from the 1970s through the 1990s! However, they suffered from cryptanalytic flaws that could be exploited by increasing levels of automation.

Alan Turing is the origin of much of what we know about computing today. You may have heard of the Turing Award, of Turing Machines or have just seen The Imitation Game... he's kind of a big deal.

The *Colossus* was a more general-purpose computing machine instrumental in breaking another rotor-based German cipher during WWII. It was the first _____, though it couldn't store its own programs: those had to be supplied via plugs and switches.

Modern (symmetric-key) cryptography

Block ciphers

Stream ciphers

Cryptographic hash functions

Random number generators

We'll discuss asymmetric-key / public-key cryptography later in the course.

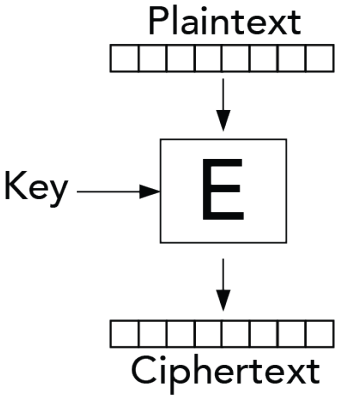
Block ciphers

Plaintext, ciphertext in *blocks*

- old DES (1970s): 64b blocks
- modern AES: 128b blocks

Shannon's principles:

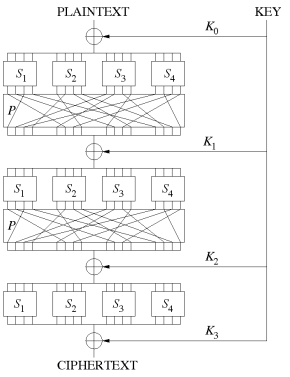
Confusion: non-linear transformations
Diffusion: changes spread



Claude Shannon is another key figure in the history of computing. He created the discipline of _____, which is why you may have heard of the *Shannon limit* in communications. He also made early contributions to modern cryptography.

SP networks

- *Substitution-Permutation* networks proposed in 1970s, still used today
- three elements:
  - S-boxes: non-linear
  - permutation: transposition
  - key schedule: subkey bits



An SP network was the foundation of the LUCIFER cipher, which became the Data Encryption Standard in the 1970s.

DES: Data Encryption Standard

Proposed by IBM

- based on *LUCIFER* algorithm

Modified by NSA

- suspicions of weakening
- evidence of strengthening

The story gets told in different ways by different people, but after the NSA got involved, DES had different S-boxes and a shorter key length. These shorter keys were seen as "adequate" for commercial uses, but made the cipher more vulnerable to brute-force attack by a sufficiently well-resourced adversary (ahem).

In 1990, Biham and Shamir published a paper on *differential cryptanlysis*, a powerful new form of cryptanalysis for attacking block ciphers. It turns out that DES was surprisingly, improbably good at resisting this form of cryptanalysis, suggesting (and later confirmed by people at IBM) that this form of cryptanalysis was known by selected people at IBM and within the NSA at least 15 years prior!

Making cryptography widely available for commercial purposes meant that ordinary people and businesses could now protect their information in ways that they never could before. This set the stage for Part I of the Crypto Wars, which we'll discuss further when we get to public-key encryption.

Advanced Encryption Standard

Replacement for aging DES in early 2000s

Open NIST competition for academic cryptographers

Winning entry: *Rijndael* algorithm

- SPN-like architecture
- 10 rounds of substition, linear mixing, key mixing
- 128b blocks, 128b/192b/256b key (AES-128, AES-192, AES-256)

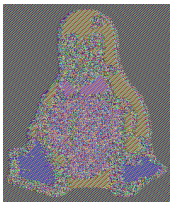
AES is so ubiquitous that CPU architectures provide dedicated silicon for it, with native instructions for encrypting and decrypting content with AES. The availability of `aesni` can have a major performance impact on applications that reply heavily on symmetric-key cryptography.

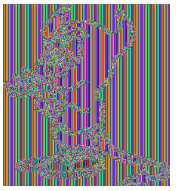
Block cipher modes

Sounds pretty secure, right?

Uhhh...

- passing the same plaintext to a block cipher with the same key will yield the same ciphertext output
- block ciphers alone lacks *semantic security*





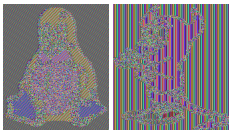
Can you tell which of these is m_0 and which is m_1?

Encrypted images generated with encrypt-image.py

Semantic security (see: Encyclopedia of Cryptography and Security, 2011 Edition, Springer) is defined as the *indistinguishability* of encryptions, i.e., an adversary cannot tell which of two candidate plaintexts has been encrypted to ciphertext.

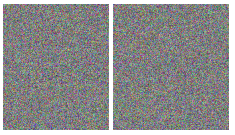
Block cipher modes

Electronic codebook (ECB) mode



- "bare" block cipher
- encrypt each chunk of plaintext directly

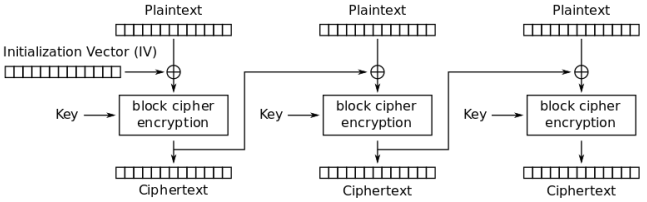
More sophisticated modes



- provide semantic security
- e.g., Cipher Block Chaining (CBC)

Block cipher modes are schemes for handling _____ blocks of plaintext and ciphertext. There are lots of modes (ECB, CBC, CTR, GCM, XTS, ...), each of which can be used with _____. So, to identify a cipher, we need more than just the _____ (e.g., AES): we also need to specify the _____. For example, AES-128-CBC is different from AES-128-GCM.

Cipher Block Chaining



Cipher Block Chaining (CBC) mode encryption

- ciphertext block depends on **all** previous blocks — *diffusion*
- result looks *really* random

Other modes

CTR and GCM modes

- used to make **stream ciphers** out of block ciphers

XTS mode

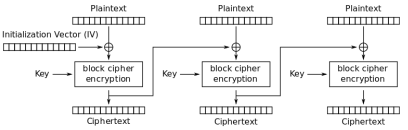
- used for full-disk encryption

... and many others ...

Message Authentication Code

What if we:

1. encrypt in CBC mode and
2. throw away most of the ciphertext?



Cipher Block Chaining (CBC) mode encryption

Message Authentication Code (MAC):

- *cryptographic* checksum that can verify message integrity **even** in the presence of an attacker (vs. checksum like CRC32)

MAC Requirements

1. Arbitrary-length message
2. Small, fixed MAC length
3. Computationally efficent
4. *Collision resistance:*
    - can't generate another message with the same MAC
    - can't generate another message with any valid MAC

Note: the Sealed Authenticator System (SAS) codes on a nuclear-armed submarine probably don't use keyed MACs, but rather purely-random codes that no human eyes have ever seen. Source: Waller, "Practicing for Doomsday", Time Magazine, 4 Mar 2001.

MAC generalization

Newer modes:

Authenticated encryption with associated data (AEAD)

What if we don't need a key?

Next time: ***cryptographic hash functions***

Summary

Classical cryptography

One-time pad

Block ciphers

Next time:

Cryptographic hash functions and passwords