

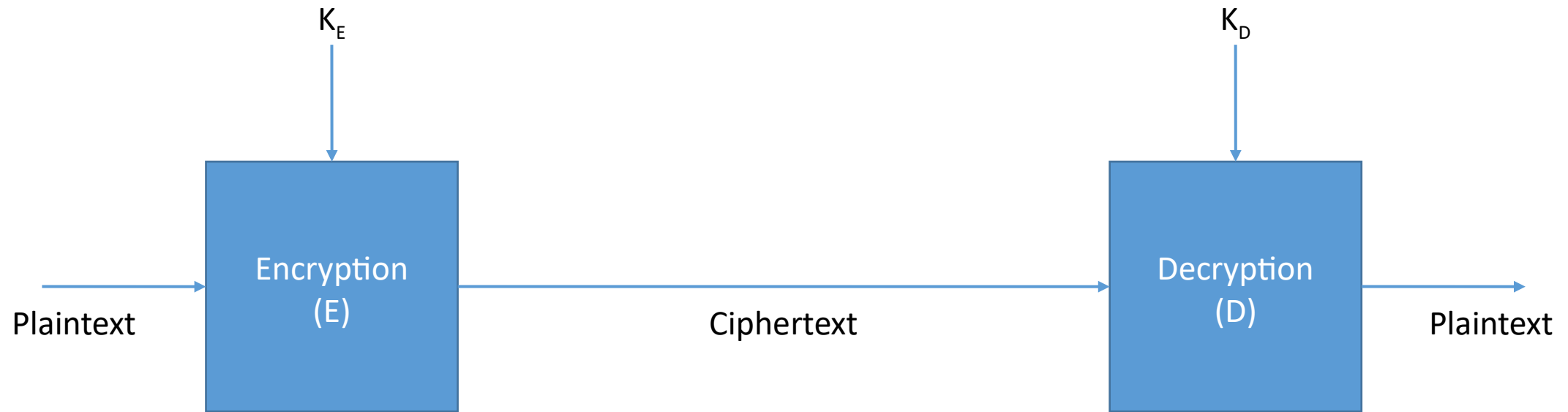
Symmetric Cryptography

Peter Chapin

Vermont Technical College

CIS-3720

Basic Concept



Symmetric Encryption: $K_E = K_D$

Notes

- Despite the word “text” the data is normally arbitrary binary data
- The key K is a small (e. g., 128 bit) binary token
- The service provided by encryption is **confidentiality**
 - Adversary is not able (hopefully!) to decode the plaintext from the ciphertext
- Encryption adds overhead
 - Time: executing E and D require CPU effort
 - Space: memory required by E and D
 - Network Bandwidth: ciphertext might be larger (but probably not)
- Key must be secure
 - Analysis assumes adversary has no knowledge of key K
 - ... but, as usual, analysis assumes adversary knows algorithms E and D

Two Approaches

- Block Ciphers...
 - ... process the input in units called blocks
 - Block size varies by encryption algorithm. Typically 64 bits (8 bytes) or 128 bits (16 bytes).
 - Creates an issue of what to do if the last block is only partially filled.
- Stream Ciphers...
 - ... process the input one byte (or even one bit) at a time.
 - No issue at the end of the input.

Block Cipher Diagram

1 block of plaintext = (for example) 64 bits



K

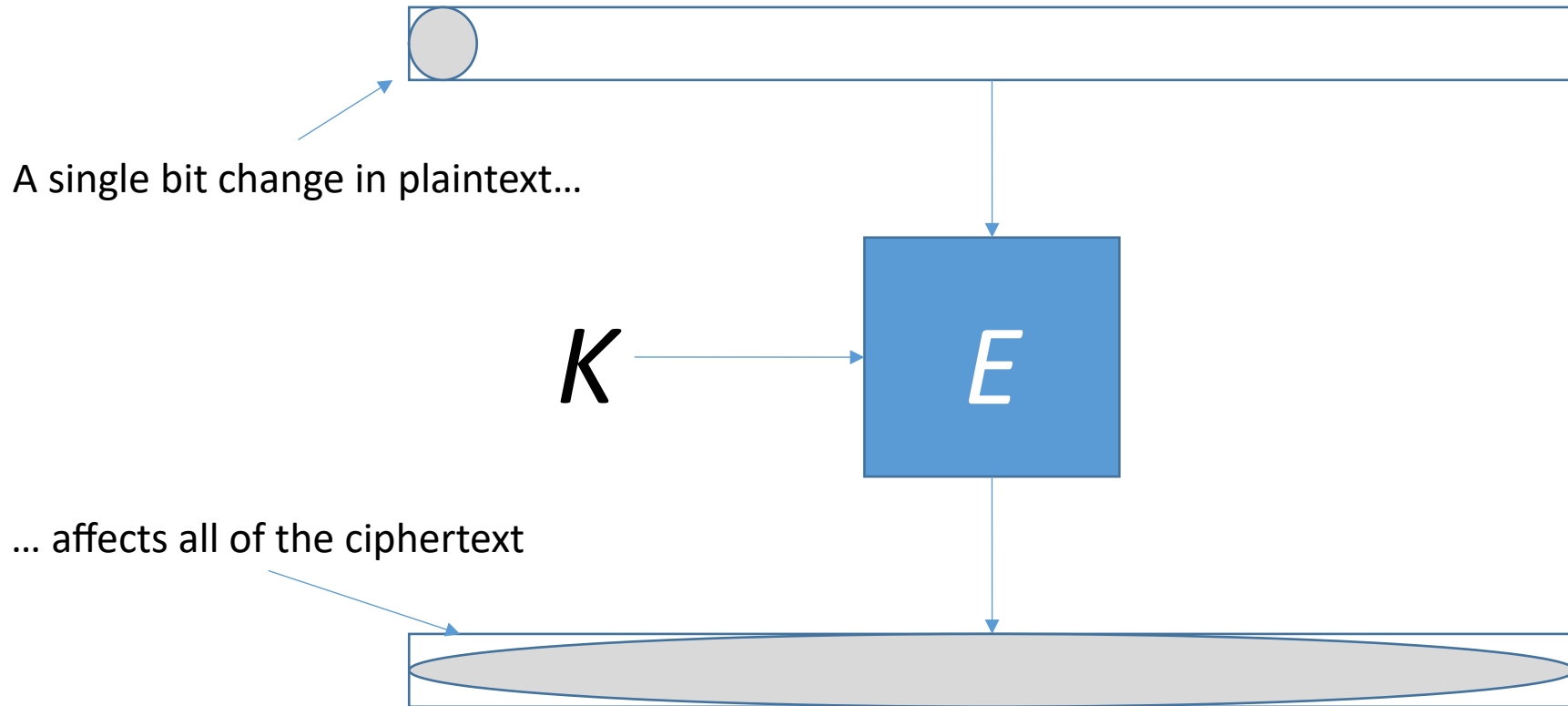


Decryption is the same picture except using D instead of E , and with ciphertext in and plaintext out.

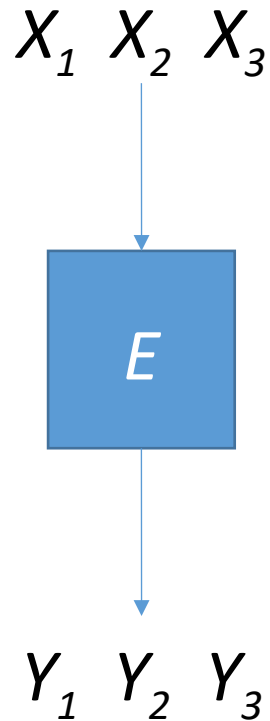


1 block of ciphertext = (as above) 64 bits

Diffusion



Confusion



Is it possible to express ciphertext bits as a *linear combination* of plaintext?

$$Y_1 = A_{11} X_1 + A_{12} X_2 + A_{13} X_3$$

$$Y_2 = A_{21} X_1 + A_{22} X_2 + A_{23} X_3$$

$$Y_3 = A_{31} X_1 + A_{32} X_2 + A_{33} X_3$$

If so, inverting matrix A allows computing plaintext from ciphertext!

If not, the encryption algorithm is *non-linear* and said to have “confusion.”

There is a matter of degree: even an approximate result can be a problem.

Note: Matrix A is key dependent, but might be computable given a (plaintext, ciphertext) pair!

Key Size?

- What size should the key be?
 - Does not have to be the same as the block size (often isn't)
 - Big keys generally take longer to process; increases overhead
 - Big keys are more secure
 - Some algorithms allow variable sized keys
- Small keys?
 - Keys less than 64 bits are not taken seriously in today's world
 - Keys of 80 bits are probably okay
 - Keys of 128 bits are more like it
 - *Where did I come up with these numbers?*

Brute Force Attack!

- Very simple: Try every possible key
 - 64 bit key implies $2^{64} = 1.8447 \times 10^{19}$ keys
- Suppose...
 - Machine can execute 10^9 trial decryptions per second
 - $1.8447 \times 10^{19} / 10^9 = 1.8447 \times 10^{10}$ seconds to exhaust key space
 - ... that's 584.55 years!
 - BUT... problem is easy to parallelize. Specialized hardware is very fast.
 - With today's technology you could probably go 100x faster
 - ... reduces time scale to ~years... or maybe ~months or ~weeks!
 - Likely to find the key when about $\frac{1}{2}$ the key space has been searched

Brute Force Attack?

- Let's use 128 bit keys
 - **128 bit key** implies $2^{128} = 3.4028 \times 10^{38}$ keys
- Suppose...
 - Machine can execute **10^9 trial decryptions per *nanosecond*!**
 - $3.4028 \times 10^{38} / 10^{18} = 3.4028 \times 10^{20}$ seconds required to exhaust key space
 - That's **1.0783×10^{13} years!**
 - That's **780 times longer than the age of the universe!**
 - Conclusion: **Brute forcing a 128 bit key is out of reach by any foreseeable tech.**

Why Use 256 Bit Keys?

- Some algorithms allow for very long keys
 - ... but long keys generally take more processing overhead. Why use them?
- Brute force effort is the threshold for cryptographic analysis
 - Algorithms are sometimes said to be “broken” if there exists a mathematical way to crack them that is more efficient than brute force.
 - ... but that doesn’t necessarily mean it is easy!
 - Attacks on (a slightly weakened form of) AES have been found that are faster than brute force,[†] but still completely impractical to execute.
- Super large keys offer insurance against future developments in cryptanalysis

[†] *Improved Cryptanalysis of Rijndael* by Niels Ferguson, et. al.

Three Way Trade-Off

- Cost
 - Super powerful computers are expensive
 - Some adversaries won't have the money to buy them
- Value of Data
 - Some secrets just aren't worth the cost
 - Adversaries are willing to spend money to learn important secrets
- Lifetime of Data
 - Daily battlefield orders useless tomorrow
 - Must be cracked quickly (which costs a lot) to be of any value

Fit Encryption Properly

- **Encryption adds overhead**
 - **Don't use more encryption than you actually need**... ideally none!
- Dial up your encryption tech to match...
 - ... the resources your adversary is likely to use against you
 - ... the value of the data you are protecting
 - ... the lifetime of the data you are protecting
- Q: What's the ideal key size?
 - A: **The smallest one you can get away with**
 - ... while making sure to protect your data for as long as needed against an adversary willing to spend an "appropriate" amount on discovering it

Symmetric Algorithms...

- DES (Data Encryption Standard)
 - Old standard, now obsolete (key too small). **Do not use!**
- AES (Advanced Encryption Standard)
 - Current standard. 128 bit block, 128 or 256 bit key
- 3DES (Triple DES)
 - Still used but slow. 64 bit block, 112 or 168 bit key
- CAST, Blowfish, Twofish, Idea, MARS, Serpent, RC5...
 - ... and a cast of others