

Symmetric Cryptography

UT CS361S

SPRING 2021

LECTURE NOTES

A solid orange horizontal bar spanning the width of the slide at the bottom.

Technology Review

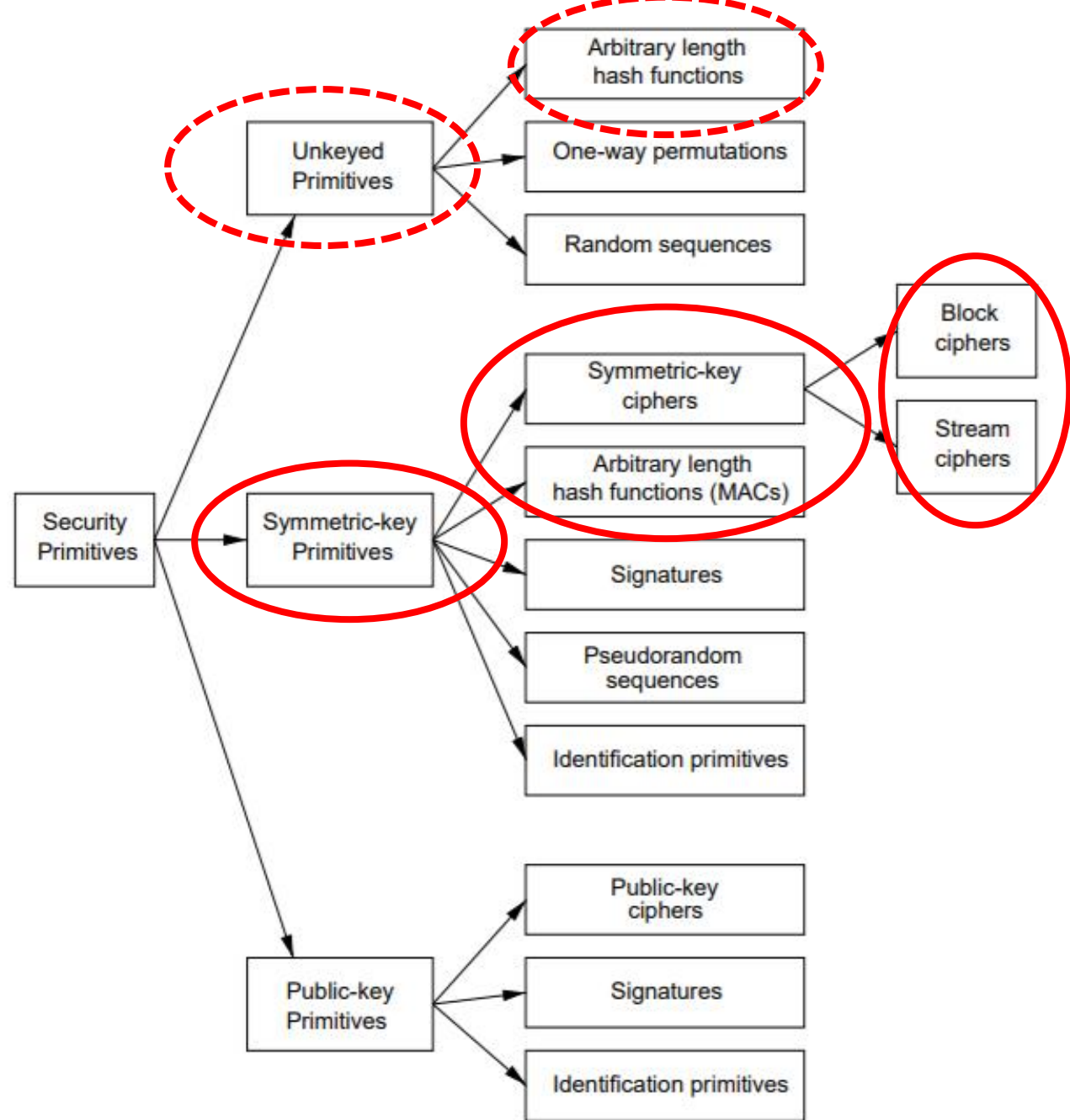


Figure 1.1: A taxonomy of cryptographic primitives.

Symmetric Cryptography

All parties in the protocol share a unique key

If $|\text{parties}|$ is 1, “secret key”

If $|\text{parties}| > 1$, “shared key”

For today’s examples, assume pre-shared key

Symmetric- Cipher Types

Block Cipher

- Data split in fixed-size blocks
- 1:1 map from plaintext block to ciphertext block
- “Substitution Cipher”

Stream Cipher

- Encrypted 1 symbol at a time
- Combined with a **Stream** of key material (key stream)

Monoalphabetic Substitution (Caesar cipher)

Each letter of plaintext maps to exactly one cipher symbol
(Block Size: 1 letter)

Let's race! Decrypt the following:

RYG WKXI CSLVSXQC NY IYE RKFO?

It's a question, when you decrypt it, shout out the answer!

Caesar Cipher: “Shift” Cipher

Write out all letters and move them over (wrapping around)

A	B	C	D	E	F	...	X	Y	Z
X	Y	Z	A	B	C	...	U	V	W

KEY SPACE: How many keys are there for this algorithm?

Permutation Cipher

Any letter can be mapped to any letter:

A	B	C	D	E	F	G	H	I	J	K	L	M
X	N	B	Y	A	M	L	S	V	P	R	K	W
N	O	P	Q	R	S	T	U	V	W	X	Y	Z
Z	C	G	I	U	D	T	F	O	H	J	Q	E

Key space is now how big?

Block size is still 1

Key Space is related to exhaustive/brute-force search

- Even without a computer, a 25/26 key space is easy to brute force

Block Size is related to *cryptanalysis* e.g., frequency analysis

- A one-letter block does not conceal enough information

Why Size Matters

Playfair Cipher

1854 by Charles Wheatstone

Named after Lord Playfair who promoted it (classic politician)

Example: <https://geeksforgeeks.org/playfair-cipher-with-examples>

Key: monarchy

Plaintext: instruments

Playfair Keyblock

5×5 grid of alphabets that acts as the key (**Key Expansion**)

One letter of the alphabet (usually J) is omitted from the table

(If the plaintext contains J, then it is replaced by I)

M	O	N	A	R
C	H	Y	B	D
E	F	G	I	K
L	P	Q	S	T
U	V	W	X	Z

Playfair Encipherment

The plaintext is split into pairs of two letters (digraphs).

If there is an odd number of letters, a Z is added to the last letter

PlainText: "instruments"

After Split: 'in' 'st' 'ru' 'me' 'nt' 'sz'

Playfair Encipherment

If both the letters are in the same column:

Take the letter below each one

Diagraph: "me"

Encrypted Text: cl

Encryption:

m -> c

e -> l

M	O	N	A	R
C	H	Y	B	D
E	F	G	I	K
L	P	Q	S	T
U	V	W	X	Z

Playfair Encipherment

If both the letters are in the same row:

Take the letter to the right of each one

Diagraph: "st"

Encrypted Text: tl

Encryption:

s -> t

t -> l

M	O	N	A	R
C	H	Y	B	D
E	F	G	I	K
L	P	Q	S	T
U	V	W	X	Z

Playfair Encipherment

If letters do not share a row or column:

Form a rectangle with the two letters and take the letters on the horizontal opposite corner of the rectangle

Diagraph: "nt"

Encrypted Text: rq

Encryption:

n -> r

t -> q

M	O	N	A	R
C	H	Y	B	D
E	F	G	I	K
L	P	Q	S	T
U	V	W	X	Z

Playfair Ciphertext:

Plain Text: "instrumentsz"

Encrypted Text: gatlmzclrqtz

How “Strong” is Playfair?

What’s the key space? How long to brute force?

What about “cryptanalysis”?

Cryptanalysis Considerations

Our example playfair ciphertext might be “uncrackable”

But what if the plaintext was longer?

How much ciphertext before **patterns** emerge?

Digraph (2-letter “blocks”) are better, but not good enough

Modern Requirements*

1. Symbols are just bits (can represent all info)
2. Blocks are large (used to be 64, now 128 bit)
3. Key size is large (currently 128 min, soon 256 bit min)
4. Algorithm has “Avalanche” property

*Not a formal list of all requirements; These are the ones that Dr. Nielson runs into most often

AES: Common Block Cipher

Block size is ***ALWAYS*** 128 bits

Key size can be 128, 192, or 256

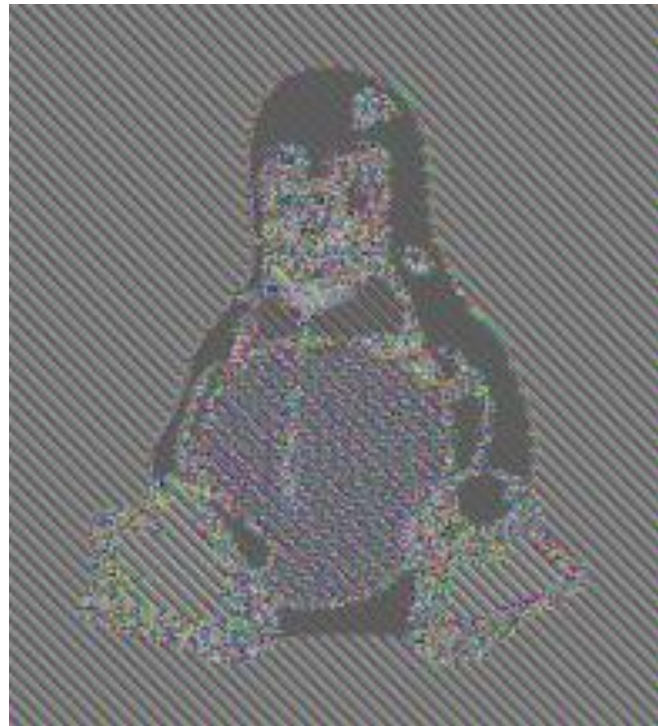
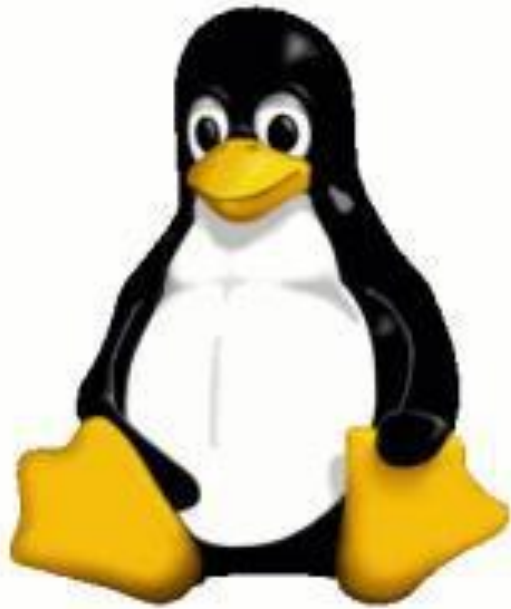
But at its core, still a “substitution” cipher.

AES-128 encryption of “My name is Seth!” w/ a “zero” key:

E94B 69E7 13C6 B4F9 B834 FEA5 95F7 8F2A


Block Cipher Weaknesses

What happened here?



Patterns

Remember, AES still 1-to-1 mapping of 128 bits



If input 128 bits is the same, output 128 bits is the same

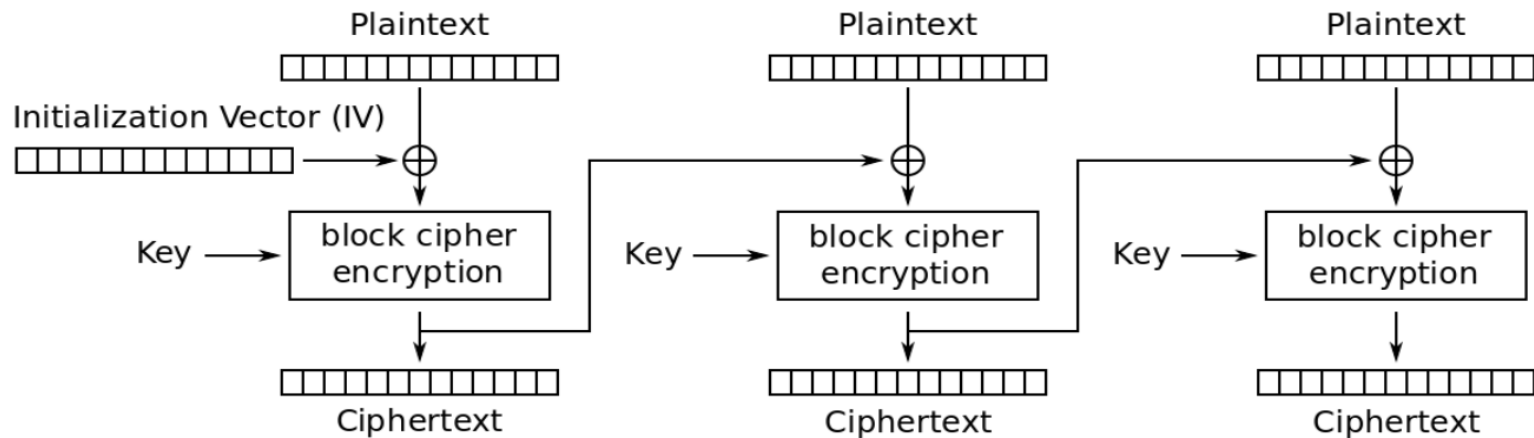


Patterns between blocks NOT ERASED



Solution: “link” output in some way

Cipher-Block Chaining (CBC)



Cipher Block Chaining (CBC) mode encryption

What is an IV?

Initialization Vector

Similar to “salting” a hash

Eliminates deterministic output

You should almost never reuse a key

BUT YOU SHOULD NEVER EVER reuse a KEY and IV

Quasi-Deprecated CBC

CBC “works” *when used correctly*

However, has various weaknesses and is error prone

In many cases, CBC is being deprecated

One-Time Pad (OTP)

Before talking about AES-Counter Mode, let's talk about OTP

OTP uses **keys that are the same size as the plaintext!**

If you have a 1GB file, you would have to have 1GB of random data!

Encryption is to XOR the plaintext with the random data

OTP has certain provable security characteristics for confidentiality

Stream ciphers mimic OTP by deriving key material from a key

Counter Mode (CTR)

Converts a block cipher into a **STREAM CIPHER**

Does NOT encrypt the plaintext directly

Rather, is used to generate a key stream

Keystream = AES(IV + 0), AES(IV + 1), AES(IV + 2)...

Ciphertext = Plaintext XOR Keystream

Plaintext = Ciphertext XOR Keystream

Don't Reuse Keystream

$$C1 = K1 \text{ xor } M1$$

$$C2 = K1 \text{ xor } M2$$

$$C1 \text{ xor } C2 = K1 \text{ xor } M1 \text{ xor } K1 \text{ xor } M2$$

$$= K1 \text{ xor } K1 \text{ xor } M1 \text{ xor } M2$$

$$= M1 \text{ xor } M2$$

Confidentiality does NOT provide Integrity

Start with a ciphertext $C1 = K \text{ XOR } M1$

Suppose an attacker knows the plaintext $M1$ (or part of it!)

Suppose attacker can intercept/change the message

Attackers wants to change $M1$ to $M2$

Attacker produces $C2 = C1 \text{ xor } (M1 \text{ xor } M2)$

$$= K \text{ xor } M1 \text{ xor } M1 \text{ xor } M2$$

$$= K \text{ xor } M2$$

ALSO WORKS ON OTP (“provably secure”)

Integrity in History

Message integrity also pre-dates modern cryptography

Bank transfers in the 19th century used the telegraph

How to keep a telegraph operator from sending a false message?

Banks developed code but this did nothing for *message integrity*

Banks developed code books with a “test key”

- The test key had one-way calculations for money, dates, currency, etc
- The test key computed and the test key transmitted had to match
- Not great by today’s standards, but worked until the 1980’s!!!

Modern Hashing

1. Compression
2. Ease of Computation
3. Preimage Resistance
4. 2nd Preimage Resistance
5. Collision Resistance

In practice, also has the Avalanche property

Message Authentication Code

MAC is a symmetric key code that is used for message integrity

Commonly implemented as a **keyed** hash

Super simple MAC: $\text{hash}(\text{message} + \text{key})$

HMAC is more complicated, but same basic idea:

$\text{HMAC}_k(M) = h(k \text{ xor } A, h(k \text{ xor } B, M))$

- A = repeated 0x36
- B = repeated 0x5c

Composite Mode

Also known as

- Authenticated Encryption or
- **AEAD – Authenticated Encryption with Additional Data**

Integrity + Confidentiality

AES-GCM is counter mode with a built-in MAC (called a “tag”)

AES-CCM is counter mode with CBC-MAC

Only AEAD ciphers supported in TLS 1.3

TAKEAWAY MESSAGE: USE AEAD WHENEVER POSSIBLE!!!!!!!