

#### Computer Network Security

COMP 178 | Spring 2025 | University of the Pacific | Jeff Shafer

# **Cryptography:** Intro, One Time Pads, Block Ciphers

# Let's talk about cryptography

# An hour ought to do it, right?



ACrJ0XSN2Jod	f5/SLFJORIBGOCZD5Y	AAYxXLxkkat	xO/Byg+Ooi						
yjIXCzW4X6s7BoBGRGoMrlcVl1xrJOPidWbd6f2rmFzwXkM2F9h									
AC70YJPjuDeC:	zB9BjKu43tDiJykpM9	XdB5iE9P8n6	7olfdGvffs						
OL6wyAwfC00L		Q5ckmqfZxd	GVH5RoEccn						
n/rkz7kł	vetography is bord	t0/NXeJDcLvS8E8tW20Z							
DBoB+xVç Cr	yptograpny is nard	OiW4uxIzJNPXMzH3ZQxI							
3kaxUfX(	· · · · · · · · · · · · · · · · · · ·	X/9egvUrYDrDztlm7p5D							
5n/QjKBRQOc7NfLhvCVk3b7GqSVprCKjjav48jR1bZqMpZaZZLs									
W447y7cbPLiv2		JGjopN	V5yVs86bfV						
18K+moU4ZjqzI	Cryptography is ha	ard <sup>I6hZ2bo</sup>	cnpkbsCokt						
sV9qAfTIMaW7		rLeW8	jxaQARAQAB						
yBWYXJlbGFzICnqiaonimiuuzzvykokoiznyaXN0b3MudmFyZWxh									
2tlci5ncj6JAjkEEwECACMFAlSIDw0CGy8HCwkIBwMCAQYVCAIJ									
QIXgAAKCRCxx:	fYVq₽j≯		RdZTsaBOO						
qBB+NYlgJD231	D9Sv9CV Cryptography	is (very) hard	3CTmdP06B						
nrtT7JlTWMpor	nudNF9c		gJWD62LZ0						
vjYczDRBu3oZKPNHPmUayvIS/A//4YZwDb3Hdg1mfTAqmV8+Bce									
bcx3UdZfscFNalS3ir7YXtqwCPdb2iB0vYJgaTwkCxtfvYyLWc2									
d5dx+kthvd08VHHaLc+o2onem/FKqeDDcA1ph0I6poBActa498g									
h9JL+hZtOFNhxlEhVNV4lJjztm2/tWYTa9oIE5sX7BjaSG0HcIo									

#### If it's **good encryption**, it should look like **random noise**

But just because it **looks** like random noise doesn't mean it's **good** encryption



#### "Trust the Math"

#### ~ Bruce Schneier

https://www.schneier.com/

### Cryptography is Harder Than It Looks



#### "Cryptography is harder than it looks"

- It looks like math, and we have smart mathematicians, so problem solved, right?
- Problem: Math equations can't secure anything
  - Write equations into software
  - Embed in a larger software system
  - Manage by an OS
  - Run on hardware
  - Connect to a network
  - Configure and operate by users
- Commonly find vulnerabilities not in underlying mathematics but in the <u>implementation</u>

https://www.schneier.com/blog/archives/2016/03/cryptography\_is.html

### Cryptography is Harder Than It Looks



### Cryptography is Harder Than It Looks



#### "Complexity is the worst enemy of security"

- More lines of code
- More interactions with other systems
- More configuration options
- **Result: More vulnerabilities!**

#### https://www.schneier.com/blog/archives/2016/03/cryptography\_is.html



### Cryptography

**Computer Network Security** 

Will try to stay above the math as much as possible...

Sar dur

do

) Gilb

(Xn tr D

Vt

ATA

#### **Useful References on Class Website**

#### Cryptography

- Crypto 101 e-book [Free/PDF, (2017+)]
  - Written by Laurens Van Houtven and distributed at <a href="https://www.crypto101.io/">https://www.crypto101.io/</a>. Updated directly at GitHub repo. (See also: Local mirror, may be out of date compared to github copy)
  - Big picture ideas + Commentary on breaking crypto. Light on the math! Incomplete / work in progress.
- Cryptography: An Introduction (3rd Edition) [Free/PDF, 2006]
  - Written by Nigel Smart, a cryptographer & computer science professor specializing in the area of elliptic curve cryptography.
  - Note: The free e-book is no longer being updated as-of 2016. See Cryptography Made Simple (2016, Springer Publishing) for purchase.
  - Heavy on the math!
- Cryptography Engineering [Paid, 2010]
  - Written by Niels Ferguson (Cryptographer, Microsoft), Bruce Schneier (Cryptographer), and Tadayoshi Kohno (Professor, University of Washington)



Spring 2025

#### +**CRYPTO**101

Crypto 101 is an introductory course on cryptography, freely available for programmers of all ages and skill levels.

Get current version (PDF)

7 5 У Tweet

#### Start to finish.

Comes with everything you need to understand complete systems such as SSL/TLS: block ciphers, stream ciphers, hash functions, message authentication codes, public key encryption, key agreement protocols, and signature algorithms.

#### Learn by doing.

Learn how to exploit common cryptographic flaws, armed with nothing but a little time and your favorite programming language.

Forge administrator cookies, recover passwords, and even backdoor your own random number generator.

#### https://www.crypto101.io/

#### Has a video too! <u>https://www.youtube.com/watch?v=3rmCGsCYJF8</u>

Works everywhere.

formats:

PDF (for Mac and PC)

Mobi (for Kindle)

DRM-free and available in all common

EPUB (for most ebook readers, iPad and iPhone)

#### XOR & One Time Pads



13

14



### Useful XOR Properties

- Associative: Apply XOR in any order:  $a \bigoplus (b \bigoplus c) = (a \bigoplus b) \bigoplus c$
- Commutative: Flip operands around:  $a \bigoplus b = b \bigoplus a$
- ✓ Useful property for encryption:
  a ⊕ b ⊕ a = a ⊕ a ⊕ b
  = 0 ⊕ b
  = b

#### One-Time Pad (OTP)

						a la serie and	and the second second second	
-	interestion to	EASTERES.	3735585B	WFERATTE	MALLAGS LG	CHESKIWE	WARASTE	
	RNR2LOYX NY32BLO3	2BH2MGEL	PDLL9N92 YZPCHER4	PBXWYTXX GZWBSBBW	2872NAQ6 BYBRCO4W	THREE 49	AZMANDEN	
	40899702	THTBJZM6	BDL2962A	AGFEZYLF	GT7BQYER	GEDBEN3#	XHERLLTR	
	PAEC43AK	Porsepre	exnorump	41.9CQ74B	F4XRAND\$	GPCD2445	AZABGXEG	
	SHXMMGRY	WEXNLERE	2QYRCD77	ROTXOGWE	FC108GG3	D78EL6E7	BPLIKIRT	
	4325543%	GYARHQQ9	DEREC 183	G3MH8C7M	NNGR26EF	72023692	LGA7CNL9	
	7.							
			0					
	LM6MCGZH	EDXNRRA 7	EXLB483G	CIDEBESH	RETACOLL	6ZR3XZE3	93CA49X8	
	DNIZKPAT	HTBAD329	RSFRIEXA	P9SA48ZG	78882826	SY2P2ARH	XEEEFL23	
	8278788G	TZARZGYA	r04863M3	RILRGRIT	PHOXNCF3	XARNHÉHE	TXXDIGIT	
	GIEN/QEN	APBTER26	BCLSTRMM	BNT7SCPA	X33MPP7Q	EQ#4QFX'S	FBAPKZXX	
	NLLP3AXE	BYASSNER	NEXALEES	PMF3TZ9F	RQINFLOG	TEMMAGEC	FAREMOXZ	
	CHARENON	E3RC42C8	45496748	9C93N39M	14326MTC	781.79302	INBCEXCO	
	ACARAXXX	<b>CACEREKE</b>	SE22BTY7	OLIGMANI,	84E288LH	GWRGNC77	NCIERSENW	
	CZAZMWZB	DEED	TXECHLEE	22LTYPTG	FGCMRMQ7	2F94CTRR	AW487NNX	
	MFYMRQB7	Pr	~G2248¥	01.284427	29676C8X	NDEXYLXE	2GX93M6B	
	/							
	9.							
	62		46BB	NEEKIMGW	CLEFKDAD	1.7406020	MODERA PH	
	*		AARE	TTTECP2E	92MYYCH7	278.449.920	NZ XAEAFR	
	A States a		BCWA	BXZEFWZK	GWERLLED	QPERAF3R	DHY EXCHANCE	
1			IDTM7	ROALXYNS	BASINHSPE	MYLBARD	XXXXDX7DX	
			6R9A3	7NAKGLAN	DEMONSREE	WBWYNCHF	LERGQLCA	
1			KEK9K	PEDERFNE	RTY230WT	NEGOTNES	NOGERXLC	
			MEYNE	DOFTGBRG	MOTZODXE	EMACP820	7ZW2R7TX	
			TELCKD	E2COGRE6	4R6Y2TPT	NSHYTCKC	OZCKECYG	
			TREAKOS	XHEDENGH	LKS7QYFD	TENSYDGE	G2K6PBYK	
				NCYDRN76	SYPEREC	RRNR488X	ST7CN96X	
			ZHCKE TON	2. 27 13 22 10 10 19 13	A22234478	AZC323FP	HK 4 QENNY	
			THARW.3NT	THE REAL PROPERTY	OCH883TB	EQCXY7K3	DAEQ4EMY	
		Carlo Martin	CHEMY3PA	KACHE KEN	G9E3H9DW	HA9Y7CWE	PREPREXA	
		×	OXDANEW2	MICTEEPA	EGAYEB3E	OFMSZEOP	WHIN 3ZGNY	
		PR	BLCFKTON	TOATRUBA	143342827	PHOBZWEP	QBKDQKXR	
		PTE	RYWCYBLD	YCKXCHNP	94107802	HEGYGLPF	2CBG8F3X	
		E DYM	ARTAGR6H	LEQESGEG	YTMAC66M	CHC736LA	JNGRPOER	
		Anterior	00745032	4NMAXAX9	A FINANCE DE	ZFLGCGQC	L'ASALONS	

Pencil and paper technique

Dates back to 1882 (Frank Miller, designing for telegraph machines)





### One-Time Pad (OTP)

- Ciphertext provides no information about the original message (beyond length)
  - Thus, OTP has property of Perfect Secrecy ☺
  - Eve has no idea if P<sub>i</sub> was 0 or 1 Cannot be broken!
- But, Perfect Secrecy property is only true if: 8
   ....

#### One-Time Pad Problems

- One-Time Pad must be used only <u>once</u>!
  - Problem: Repeated use allows statistical comparisons easy to break
- One-Time Pad must be (cryptographically) random
  - Problem: Random number generation is hard/slow
- One-Time Pad must be shared between parties
  - Problem: Sharing must occur in advance of communication, out of band?
  - Both parties must keep pad secret
- One-Time Pad must equal length of plaintext
  - **7** Problem: Length!

#### Beyond the One-Time Pad

#### → What do we use instead? Ciphers!

- ↗ Manageable key sizes ☺
- Methods to negotiate keys over the Internet with parties we've never communicated with before <sup>(C)</sup>
- Systems are more complex and lack the theoretical Perfect Secrecy property of OTPs 😕





**Computer Network Security** 

- Encryption function E
  C = E(k, P)
- Decryption function D P = D(k, C)
- Symmetric-key encryption
  - **オ** Same key is used for both encryption and decryption
- Operates not bit-by-bit but block-by-block
  - Block is a collection of bits
  - **ℬ** Block size is *fixed* by the cipher



- What's in the box?
- Keyed Permutation
  - Permutation every input block is mapped to a unique output block
  - ↗ Keyed The key determines the exact mapping
    - Mapping varies by the key

### Bijection



#### Bijective function

- Function between the elements of two sets
  - Each element of set X is paired with exactly one element of set Y
  - Each element of set Y is paired with exactly one element of set X
  - There are no unpaired elements

#### Example Block Cipher

**3** bit block size (tiny!)



- Assuming no fatal flaws in mapping, brute forcing key is only path for attacker
  - Must try every potential key
  - **7**  $2^3 = 8$  possible blocks
  - **7** 2<sup>128</sup> = 3.4x10<sup>38</sup> possible blocks
- *n*! permutations of an *n* element set
  - 128 bit key?
     (2<sup>128</sup>)! permutations

#### Pros

- Key are short / fixed length
  - **7** 128-256 bits
- Much easier to communicate to recipient than a One-Time Pad!

#### Cons

- What happens if the plaintext is larger than the block size?
  - Example: AES block size is 128 bits (16 bytes)
- How do the sender and receive agree on the same key over an insecure channel?

### Block Cipher Examples

- AES Advanced Encryption Standard
- Blowfish/Twofish/Threefish
- → DES/3DES

### Block Cipher Examples - AES

- Public peer-reviewed competition to select next-generation block cipher (AES)
  - Sponsored by NIST, 1997-2001
  - Evaluated on: security, licensing, implementation
- **Security** 
  - "The extent to which the algorithm output is indistinguishable from a random permutation on the input block"
  - Soundness of the mathematical basis for the algorithm's security"
  - "Other security factors raised by the public during the evaluation process, including any attacks which demonstrate that the actual security of the algorithm is less than the strength claimed by the submitter"
- Licensing (worldwide, non-exclusive, royalty-free)

#### Algorithm / Implementation

- **7** Flexibility (e.g., additional key sizes, additional block sizes, stream cipher, ...)
- **7** Computational efficiency / Memory requirements
- Hardware and software implementation suitability

#### https://competitions.cr.yp.to/aes.html

### Block Cipher Examples - AES

#### AES – Advanced Encryption Standard

- Winning cipher: Rijndael (pronounced "Rhine Dahl")
  - Developed by two Belgian cryptographers: Vincent Rijmen and Joan Daemen
  - Block size: 128 bits
  - **Key size: 128, 192, 256 bits**
- ✓ Standardized: FIPS PUB 197 and ISO/IEC 18033-3

### Block Cipher Examples - \*Fish



- Three algorithms created by Bruce Schneier
  - Blowfish (1993) Don't use!
- **Twofish** (1998)
  - One of five finalists in the AES competition
  - "I have nothing but good things to say about NIST and the AES process."
    - ~ Bruce Schneier
- Threefish (2008)

### Block Cipher Examples – DES/3DES

- DES Data Encryption Standard <u>Don't use</u>!
  - Designed by IBM in 1975
  - Algorithm is "basically sound" (unless you have a large number of plaintexts to feed through system), but 56 bit key is too small on modern hardware
    - → Brute forced in under a day using 1999 hardware
  - → Withdrawn as standard by NIST (Replaced by AES)
- **3DES** ("triple DES") − <u>Don't use</u>!
  - **7** Run DES three times....
  - - Only bother if dealing with <u>legacy systems</u> ⊗

# Breaking Cryptography (AES/DES)

34

# Breaking Cryptography

#### General methods

- **Brute force** 
  - 🛪 Time / \$\$\$
  - Cryptographic "break" is any attack *faster* than brute force
- - Reduce the search space for brute force attacks
  - Make repeated observations and intelligent guesses about keys
- Attack the implementation ("Smart engineers")
  - Side channel attacks (Cache timing, power usage, software-initiated attacks like rowhammer, ...)



#### THE WORLD'S FASTEST DES CRACKER

In 1998 the <u>Electronic Frontier Foundation</u> built the <u>EFF DES Cracker</u>. It cost around \$250,000 and involved making 1,856 custom chips and 29 circuit boards, all housed in 6 chassis, and took around 9 days to exhaust the keyspace. Today, with the advent of <u>Field Pro-</u> <u>grammable Gate Arrays (FPGAs</u>), we've built a system with 48 <u>Virtex-</u> <u>6 LX240Ts</u> which can exhaust the keyspace in around 26 hours, and have provided it for the research community to use. Our hope is that this will better demonstrate the insecurity of DES and move people to adopt more secure modern encryption standards.

**GET CRACKING** 

https://crack.sh/

36

### Breaking Crypto – DES with crack.sh

- 48 Xilinx Virtex-6 LX240T FPGAs
  - **40 DES cores**
  - **7** 400MHz
  - **7** 16x10<sup>9</sup> keys/sec/FPGA
  - **7** 768x10<sup>9</sup> keys/sec total
- Exhaustively search 56-bit
   DES key-space in:
   2<sup>56</sup> / 768,000,000,000
   = ~26 hours

- Formats supported (Single-DES)
  - NETLM/NETNTLMv1 Authentication
  - **PPTP VPNs**
  - **オ** WPA-Enterprise
  - des\_crypt() Hashes

  - ↗ Known Plaintext DES
- Price: \$20-\$100

# Breaking Crypto - AES

- AES cipher has been "broken"
  - **7** Key recovery faster than brute force by a factor of 4
    - 128 bit key now offers security of 126 bits
    - 256 bit key now offers security of 254
  - **オ** Good math work, but yawn...

    - **7** Time to brute force now:  $\infty / 4$

#### **Research paper**

A. Bogdanov, D. Khovratovich, C. Rechberger, "Biclique Cryptanalysis of the Full AES". IACR Cryptology ePrint Archive. 344-371. 10.1007/978-3-642-25385-0\_19, **2011** 

# Breaking Crypto - AES

- - Demonstrated recovery of the full AES key given only about 6 - 7 blocks of plaintext or ciphertext
  - Requires spy process running on same machine as AES software. No privilege escalation required.
    - Side channel attack via processor cache timing
    - Runs in Python in under a minute!

#### **Research paper**

A. C, R. P. Giri and B. Menezes, "Highly Efficient Algorithms for AES Key Retrieval in Cache Access Attacks," *2016 IEEE European Symposium on Security and Privacy (EuroS&P)*, Saarbrucken, **2016** 

#### Side Channel Attack on AES



#### Countermeasures

- Use hardware AES acceleration
  - Hardware method does not use memory-resident tables for cipher
  - AES-NI instruction set in x86-64

Have I mentioned that writing robust crypto libraries is hard?

- Use a cipher and implementation that is resistant to timing attacks
  - Idea: All operations should take constant time
    - No conditional branches with conditions based on secret information.
    - All loop counts are predictable in advance
    - No array lookups with indices based on secret information. The pattern of memory access must be predictable in advance
  - ↗ Like NaCL library...

41