Hybrid Cryptography: File Encryption with AES and RSA

Jordan Doyle

Why Cryptography?

Confidentiality	Integrity
Information is kept private	Data is unaltered
Authentication	**Non-repudiation**
Verifying the identity of the sender/receiver	Neither party can deny sending a message

Goal: Implement AES and RSA to encrypt entire files

Without the use of libraries

Step 1: AES

Primary Steps for AES-128

KeyExpansion

SubBytes

ShiftRows

MixColumns

AddRoundKey

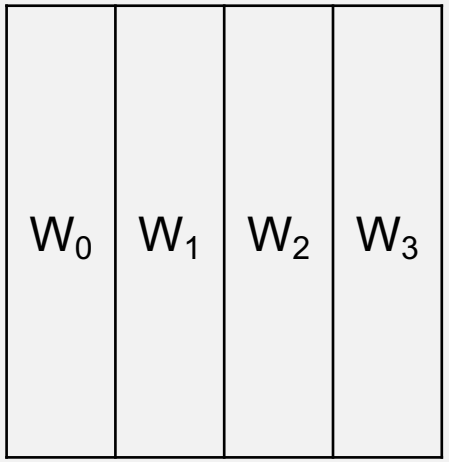
KeyExpansion

Purpose: Expand initial key into round keys

Steps:

1. RotWord: rotate the bytes in a word
2. SubWord: Substitute each byte using an S-box
3. Rcon: XOR the word with a round constant

x 10



4-word key -> 11 round keys

SubBytes

Purpose: Substitute each byte with another byte using an S-box

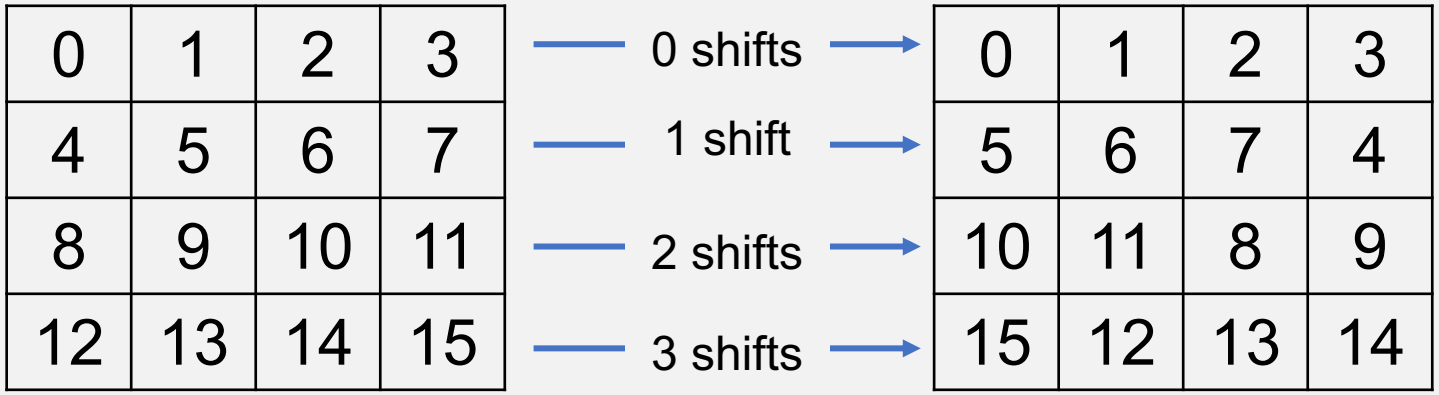
5A -> BE

	00	01	02	03	04	05	06	07	08	09	0a	0b	0c	0d	0e	0f
00	63	7c	77	7b	f2	6b	6f	c5	30	01	67	2b	fe	d7	ab	76
10	ca	82	c9	7d	fa	59	47	f0	ad	d4	a2	af	9c	a4	72	c0
20	b7	fd	93	26	36	3f	f7	cc	34	a5	e5	f1	71	d8	31	15
30	04	c7	23	c3	18	96	05	9a	07	12	80	e2	eb	27	b2	75
40	09	83	2c	1a	1b	6e	5a	a0	52	3b	d6	b3	29	e3	2f	84
50	53	d1	00	ed	20	fc	b1	5b	6a	cb	be	39	4a	4c	58	cf
60	d0	ef	aa	fb	43	4d	33	85	45	f9	02	7f	50	3c	9f	a8
70	51	a3	40	8f	92	9d	38	f5	bc	b6	da	21	10	ff	f3	d2
80	cd	0c	13	ec	5f	97	44	17	c4	a7	7e	3d	64	5d	19	73
90	60	81	4f	dc	22	2a	90	88	46	ee	b8	14	de	5e	0b	db
a0	e0	32	3a	0a	49	06	24	5c	c2	d3	ac	62	91	95	e4	79
b0	e7	c8	37	6d	8d	d5	4e	a9	6c	56	f4	ea	65	7a	ae	08
c0	ba	78	25	2e	1c	a6	b4	c6	e8	dd	74	1f	4b	bd	8b	8a
d0	70	3e	b5	66	48	03	f6	0e	61	35	57	b9	86	c1	1d	9e
e0	e1	f8	98	11	69	d9	8e	94	9b	1e	87	e9	ce	55	28	df
f0	8c	a1	89	0d	bf	e6	42	68	41	99	2d	0f	b0	54	bb	16

Image Source: Wikipedia

ShiftRows

Purpose: Perform a left-circular shift of each row



MixColumns

Purpose: Multiply each column by a preset matrix

$s_{0,0}$	$s_{0,1}$	$s_{0,2}$	$s_{0,3}$
$s_{1,0}$	$s_{1,1}$	$s_{1,2}$	$s_{1,3}$
$s_{2,0}$	$s_{2,1}$	$s_{2,2}$	$s_{2,3}$
$s_{3,0}$	$s_{3,1}$	$s_{3,2}$	$s_{3,3}$

$s'_{0,0}$	$s'_{0,1}$	$s'_{0,2}$	$s'_{0,3}$
$s'_{1,0}$	$s'_{1,1}$	$s'_{1,2}$	$s'_{1,3}$
$s'_{2,0}$	$s'_{2,1}$	$s'_{2,2}$	$s'_{2,3}$
$s'_{3,0}$	$s'_{3,1}$	$s'_{3,2}$	$s'_{3,3}$

02	03	01	01
01	02	03	01
01	01	02	03
03	01	01	02

$*$

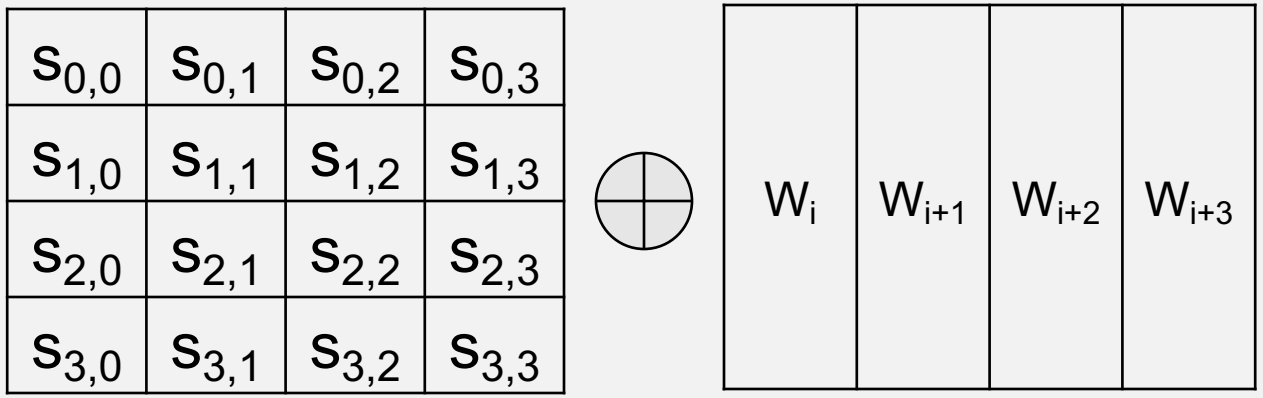
$s_{0,c}$
$s_{1,c}$
$s_{2,c}$
$s_{3,c}$

$=$

$s'_{0,c}$
$s'_{1,c}$
$s'_{2,c}$
$s'_{3,c}$

AddRoundKey

Purpose: Combine round key with state

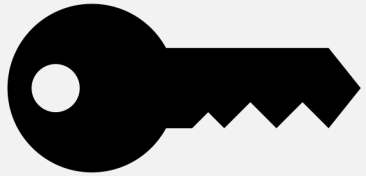


XOR state with word of round key

Extension to a file

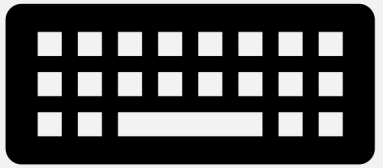
1. Read bytes from file

2. Break file into blocks



3. Encrypt

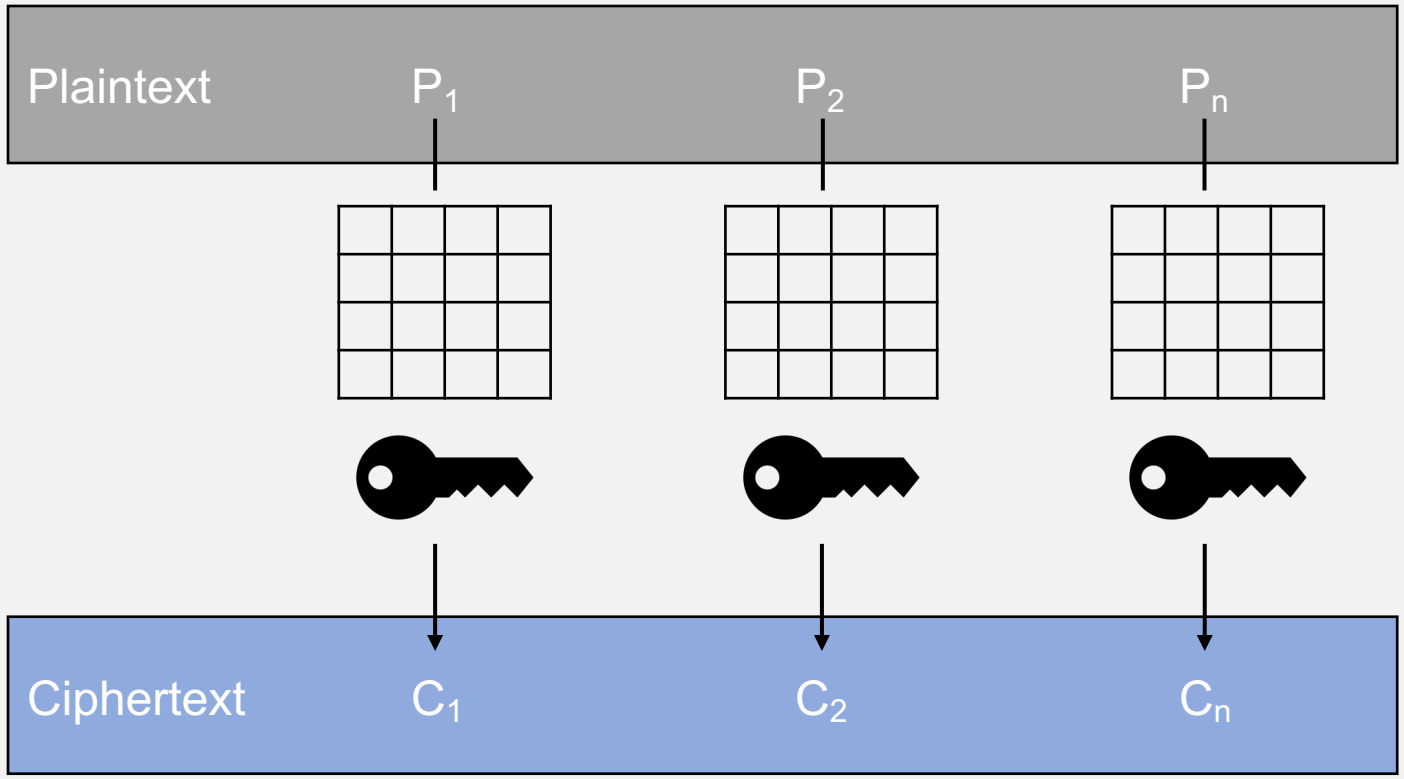
...



4. Write encrypted bytes

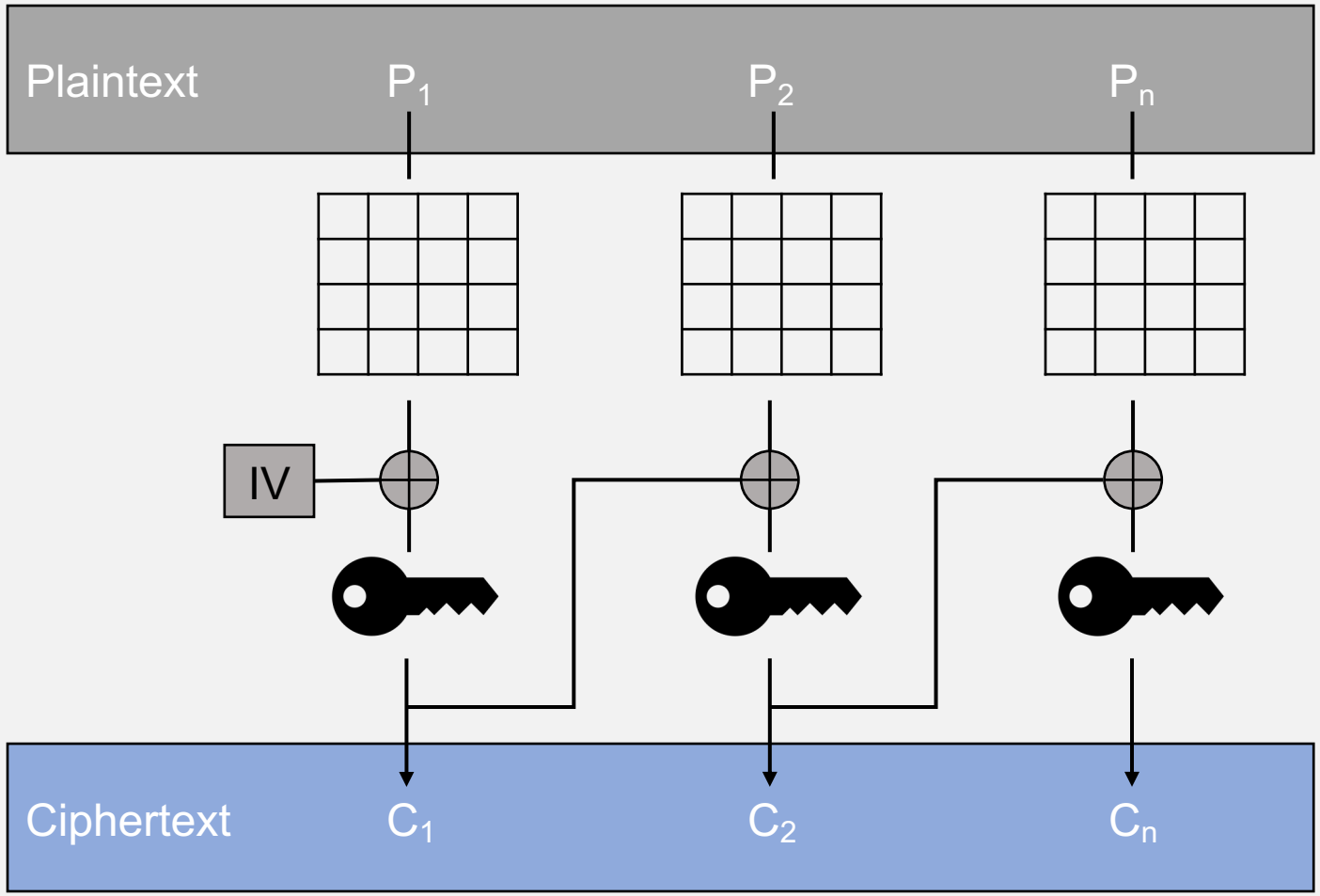
(reverse to decrypt)

ECB Mode



- Encrypt each block separately
- Combine blocks

CBC Mode



- Add padding
- XOR with IV
- Encrypt first block
- XOR encryption of previous block with plaintext of next block
- Encrypt next block

Padding – PKCS#7

- Fill block with value of the number of blocks to be filled
  - If block is full, add 16 bytes to the end

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
25	D5	A0	FA	32	60	0A	33	45	FB	96					

16 – (length of block % 16)
16 – (11 % 16)
16 – 11 = 5

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
25	D5	A0	FA	32	60	0A	33	45	FB	96	05	05	05	05	05

Extension to a file (with padding and CBC)

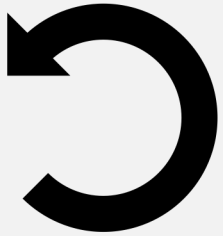
1. Read bytes from file

2. Break file into blocks

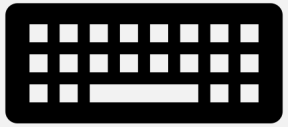
3. Add padding

4. XOR with previous block

5. Encrypt



6. Update previous block



7. Write encrypted bytes

Step 2: RSA

Key Generation - Example

p = 619

q = 199

n = 123181

p*q

Φ(n) = 122364

(p-1)*(q-1)

Key Generation: Rules for e (public key)

1. $1 < e < \Phi(n)$
2. e is coprime with $\Phi(n)$

Often pre-chosen value such as 65537

2 ✗ gcd(2, 122364) = 2

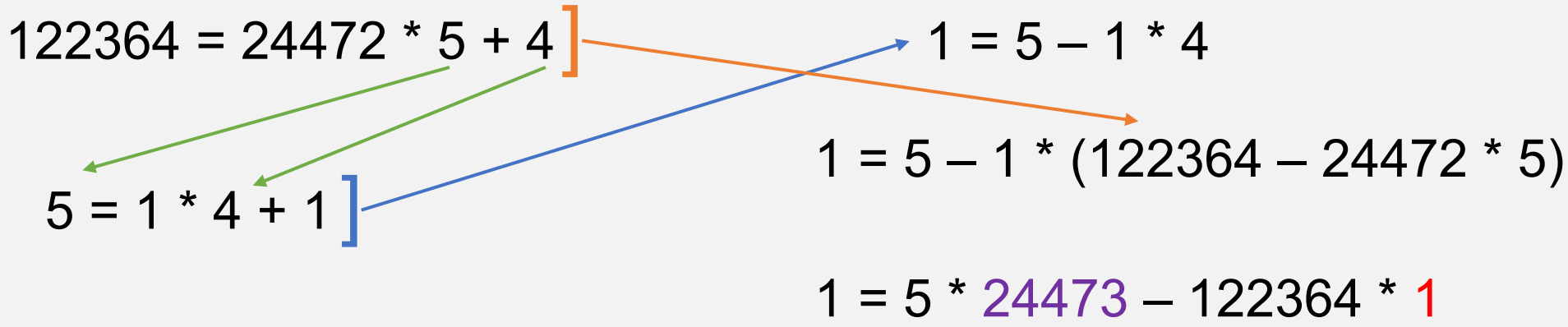
3 ✗ gcd(3, 122364) = 3

4 ✗ gcd(4, 122364) = 4

5 ✓ gcd(5, 122364) = 1

Key Generation: Rule for d (private key)

$$(e * d) \equiv 1 \bmod \Phi(n)$$
$$(e * d) \bmod \Phi(n) = 1$$
$$e * d + \Phi(n) * y = 1$$
$$5 * d + 122364 * y = 1$$



Encrypting and Decrypting Messages

Encryption

$C = M^e \bmod n$

M = 62

C = 35735

e = 5
n = 123181

Decryption

$M = C^d \bmod n$

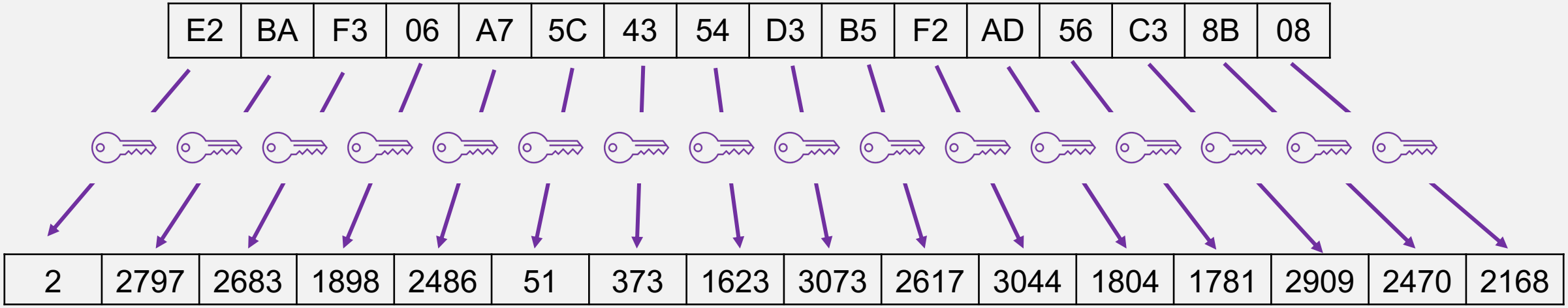
M = 62

C = 35735

d = 24473
n = 123181

Extension to Encrypt AES Symmetric Key

1. Encrypt each byte of key separately



Extension to Encrypt AES Symmetric Key

2. Use large enough primes to encrypt key all together

E2	BA	F3	06	A7	5C	43	54	D3	B5	F2	AD	56	C3	8B	08

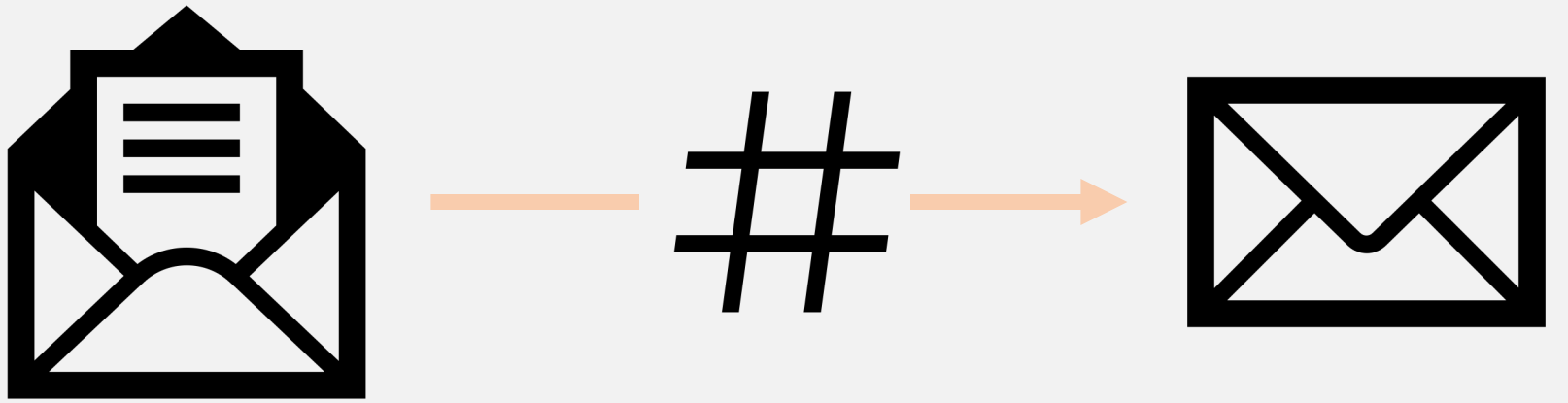
b'\xe2\xba\xf3\x06\xa7\\CT\xd3\xb5\xf2\xadV\xc3\x8b\x08'

b'\x08+\xd7\xd2\xc9\xdb\xb4\xe8\xb4CG\x9f\x85\x16i\x8d\xc5\x05?\x19\x0fB\xef\x18\xd3'

Digital Signatures

Authenticate the identity of the sender and guarantee the integrity of the message

Digital Signatures: Hash

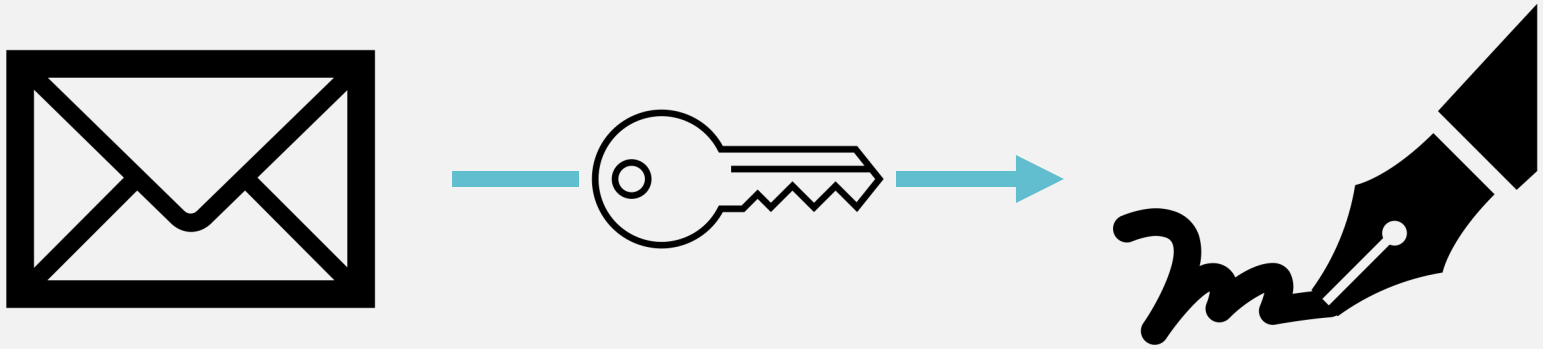


Message

SHA

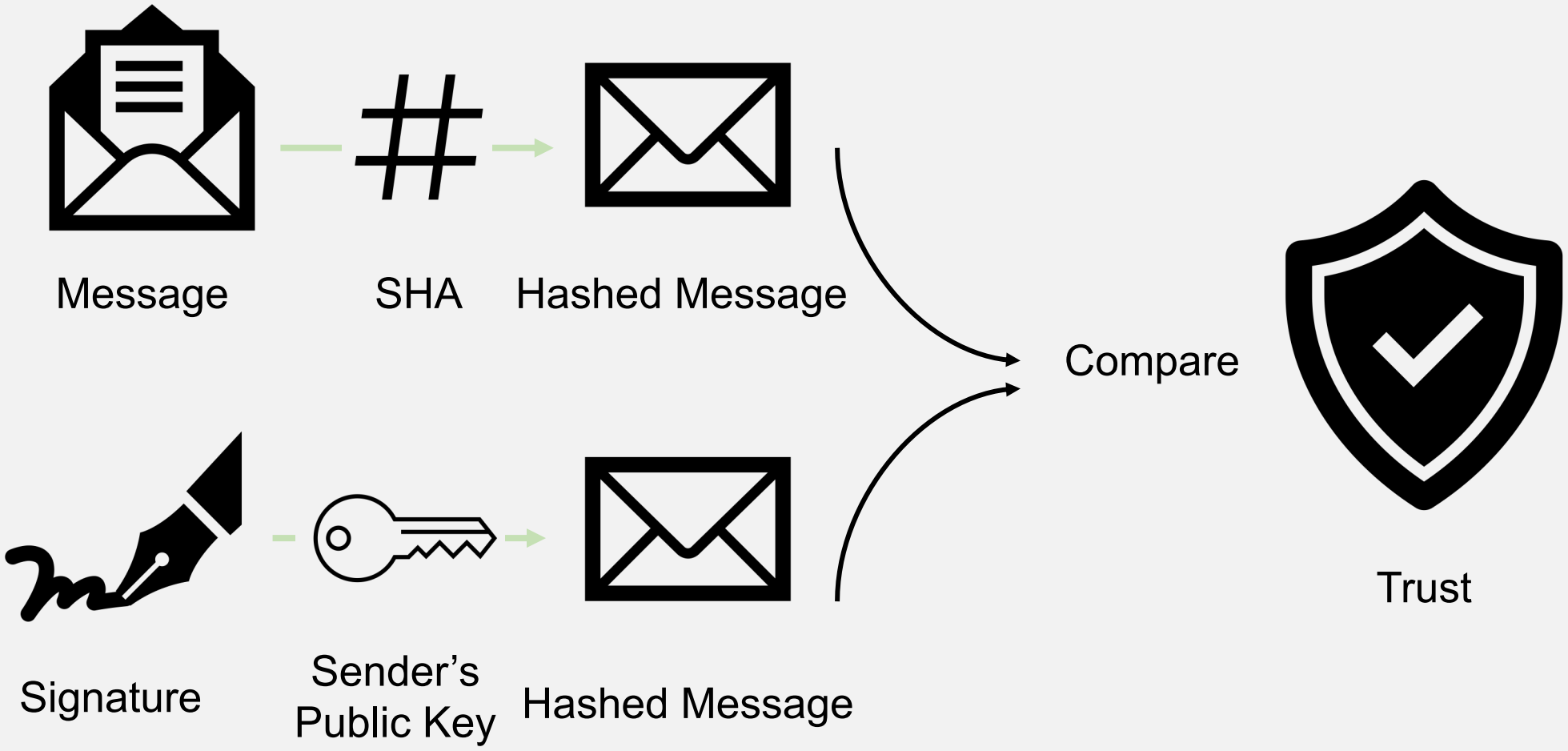
Hashed Message

Digital Signatures: Sign



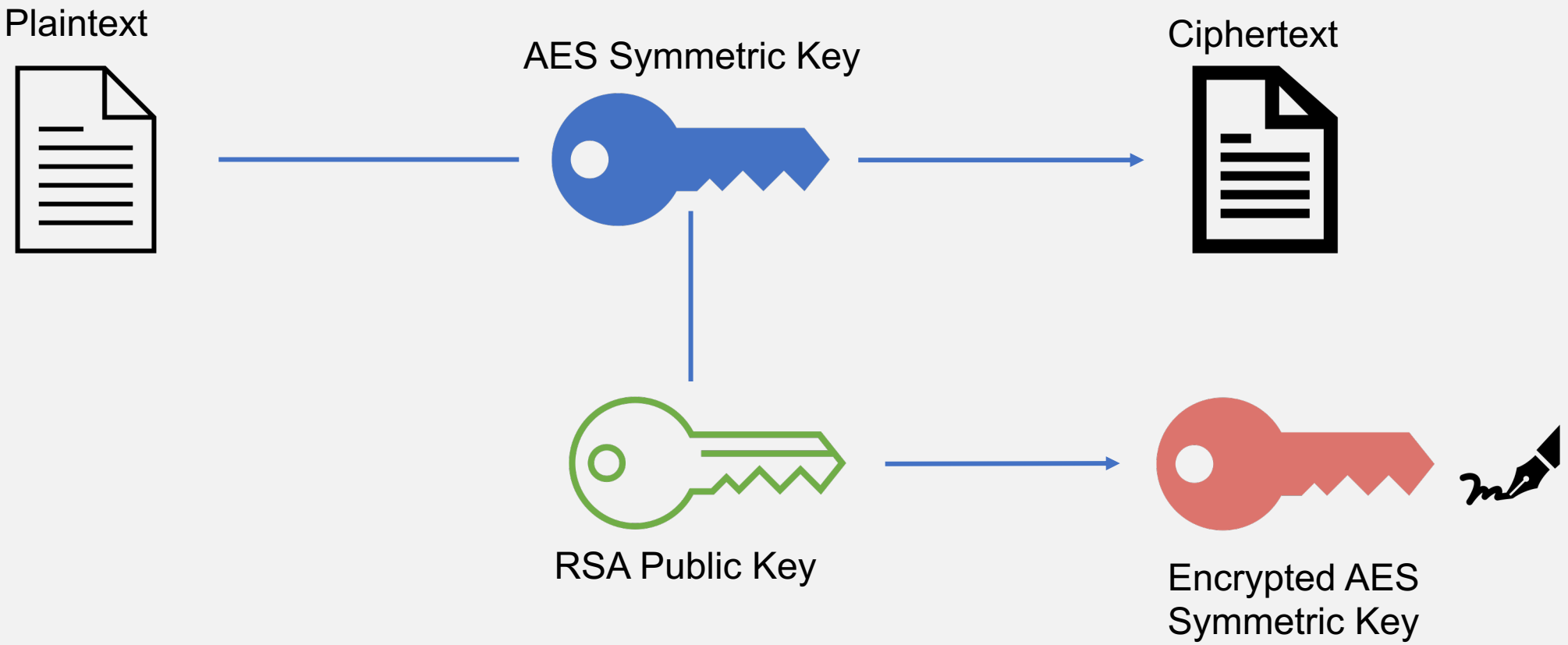
Hashed Message Sender's Private Key Signature

Digital Signatures: Verify

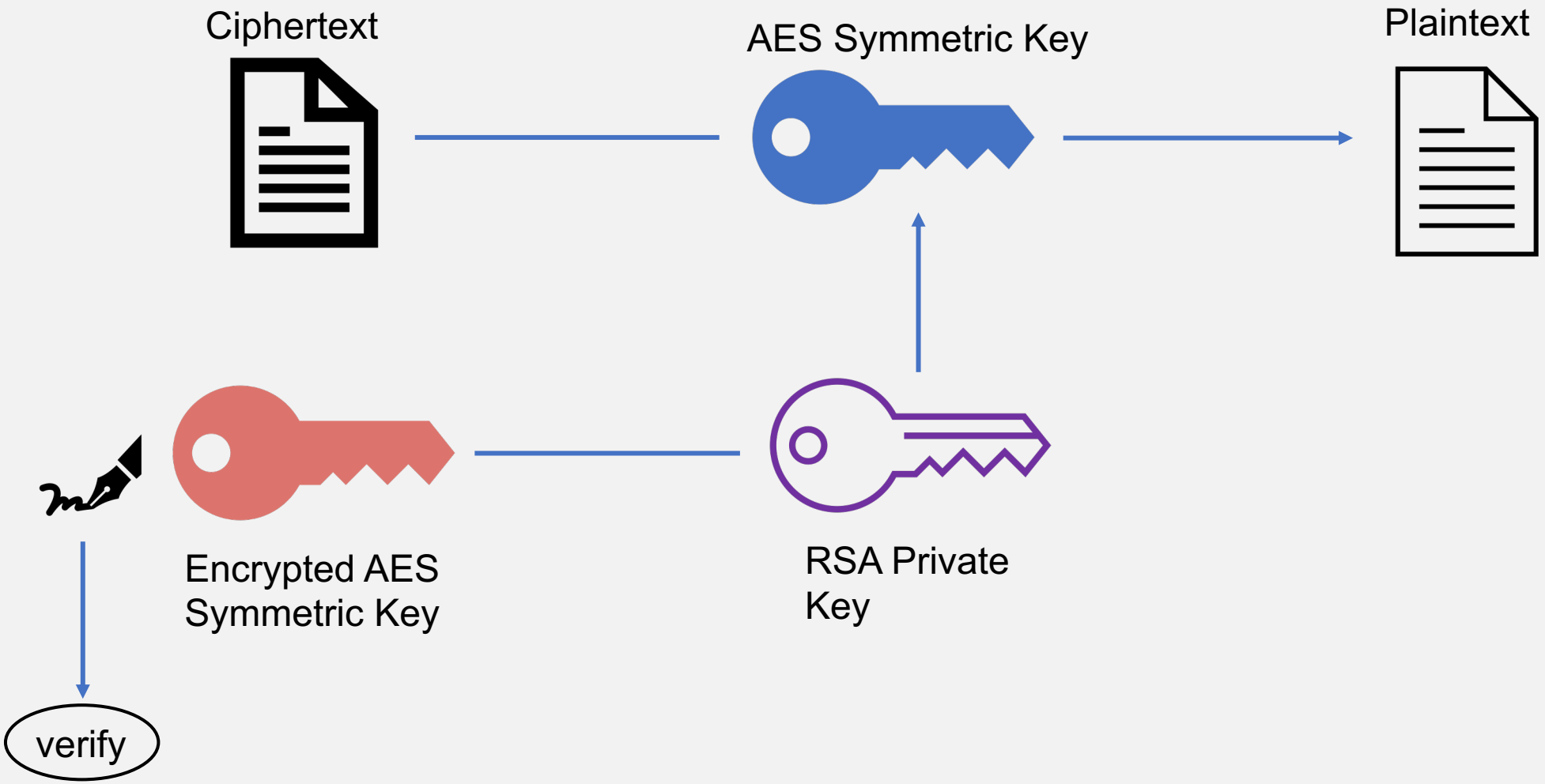


Step 3: Combine

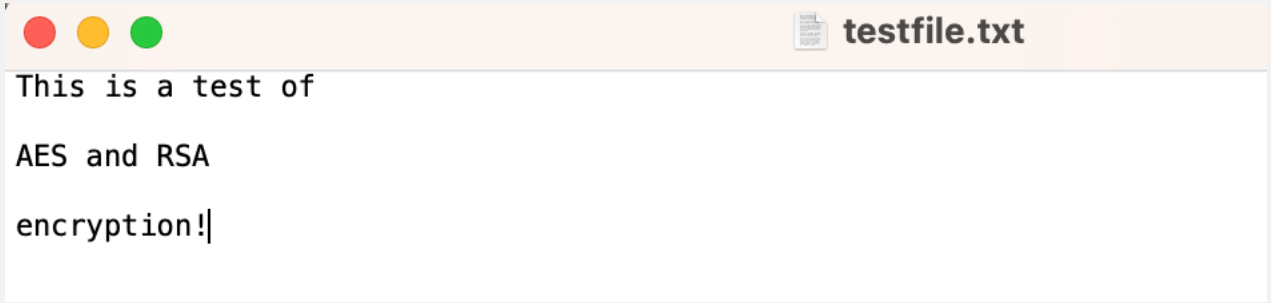
Encrypt


Plaintext
AES Symmetric Key
Ciphertext
RSA Public Key
Encrypted AES
Symmetric Key

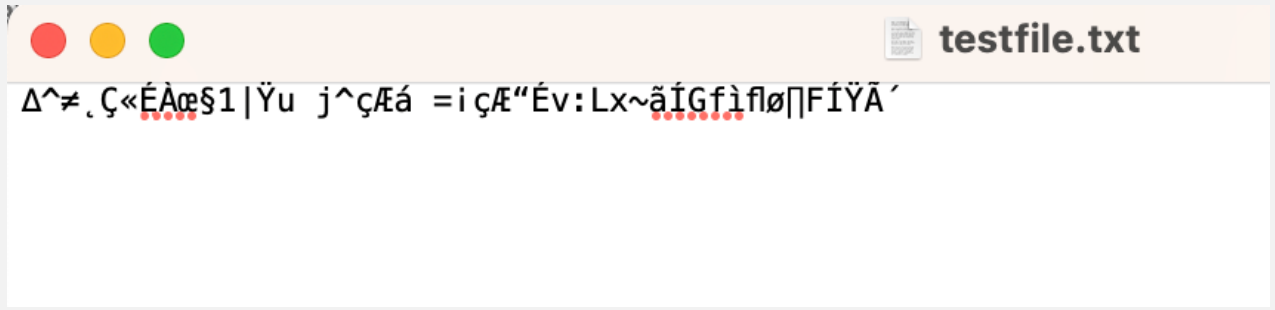
Decrypt


Ciphertext
AES Symmetric Key
Plaintext
Encrypted AES
Symmetric Key
RSA Private
Key
verify

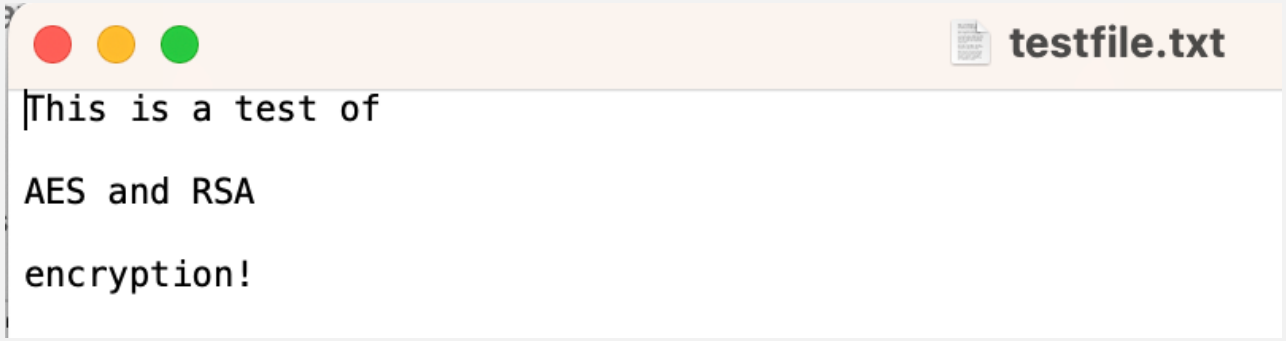
Message:



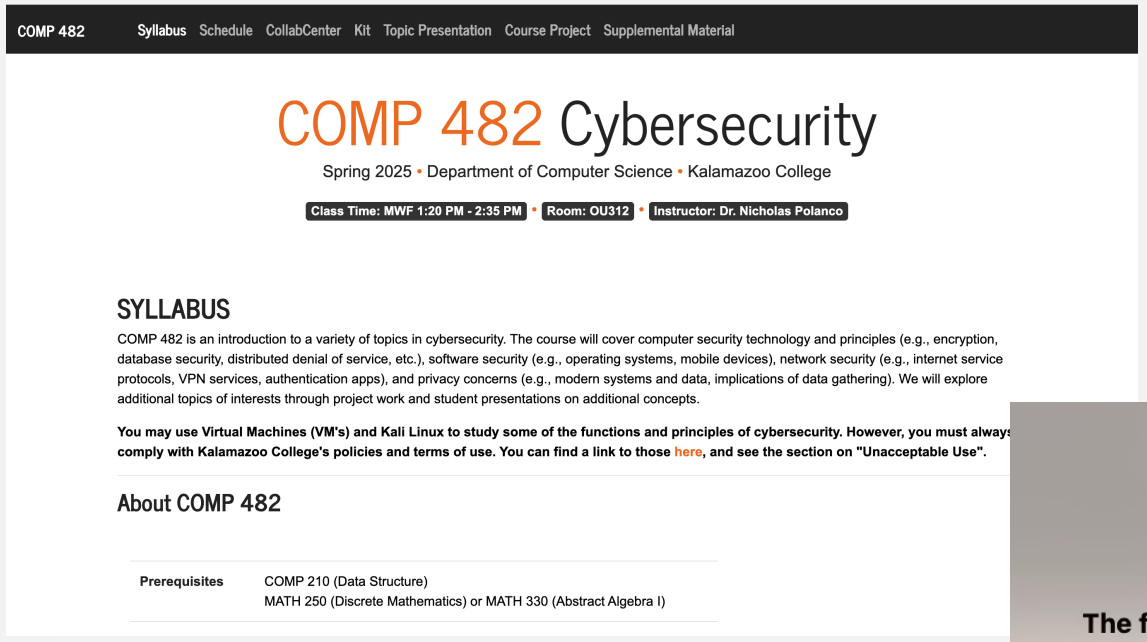
Encrypt



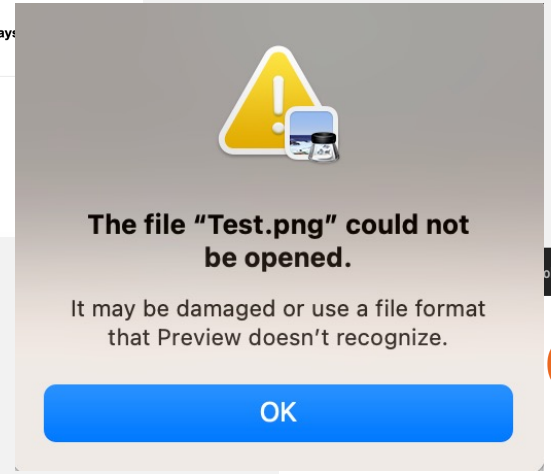
Decrypt



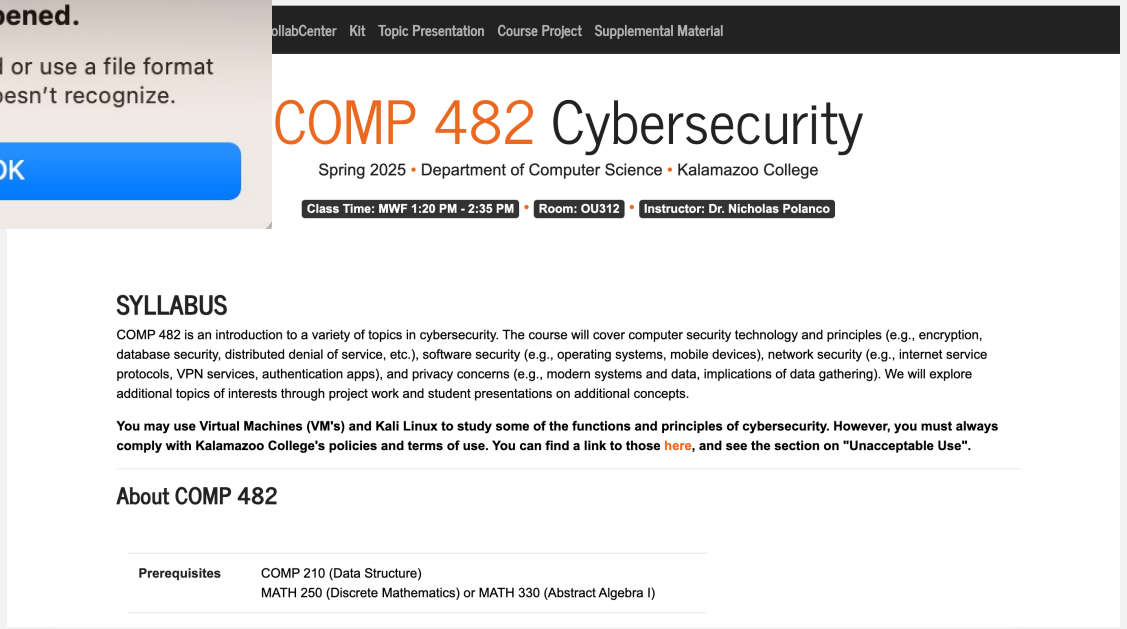
Message:



Encrypt



Decrypt



Shortfalls

- Not using large primes or generating them
- Pseudorandom values for AES symmetric key
- Not efficient

Challenges

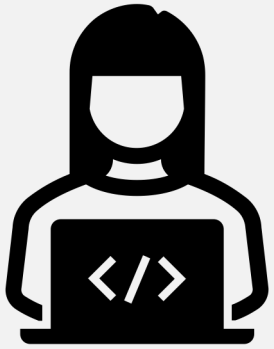
1. Multiplication in $\mathrm{GF}(2^8)$

2. CBC and Padding

3. Required length of n

Opportunities to Extend

Optimizing implementation



Complete full PKCS#7

Generating primes and randomness

My Takeaways

The algorithms behind these methods of encryption are not as complex as I would have guessed

Encryption algorithms capitalize on computers' inefficiencies

Just as we have replaced old methods of encryption, we need to continue to advance research in cryptographic algorithms

Thank You!

Questions?

Works Cited

"Advanced Encryption Standard (AES)." *GeeksforGeeks*, 3 Feb. 2025, www.geeksforgeeks.org/advanced-encryption-standard-aes/. Accessed 1 June 2025.

AES Example.

"AES-CBC Padding Explained." *Thinkinginbytes.com*, 2024, thinkinginbytes.com/posts/aes-cbc-padding-explained/. Accessed 1 June 2025.

Evans, Donald, et al. "FIPS 197 Federal Information Processing Standards Publication Advanced Encryption Standard (AES)." *Advanced Encryption Standard (AES)*, 26 Nov. 2001, nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.197-upd1.pdf, https://doi.org/10.6028/NIST.FIPS.197-upd1.

"Rijndael S-Box." *Wikipedia*, 11 May 2020, en.wikipedia.org/wiki/Rijndael_S-box. Accessed 1 June 2025.

Rivest, R. L., et al. "A Method for Obtaining Digital Signatures and Public-Key Cryptosystems." *Communications of the ACM*, vol. 21, no. 2, 1 Feb. 1978, pp. 120–126, people.csail.mit.edu/rivest/Rsapaper.pdf, https://doi.org/10.1145/359340.359342. Accessed 1 June 2025.

"RSA Algorithm in Cryptography." *GeeksforGeeks*, 6 Jan. 2025, www.geeksforgeeks.org/rsa-algorithm-cryptography/. Accessed 1 June 2025.

"RSA and Digital Signatures." *GeeksforGeeks*, 30 Dec. 2020, www.geeksforgeeks.org/rsa-and-digital-signatures/. Accessed 1 June 2025.

Slonopas, Dr. Andre. "Cybersecurity and Cryptography: Their Eternal Relationship." *American Military University*, American Military University (AMU), 3 Feb. 2025, www.amu.apus.edu/area-of-study/information-technology/resources/cybersecurity-and-cryptography/. Accessed 1 June 2025.

Wang, Shawn. "The Difference in Five Modes in the AES Encryption Algorithm - Highgo Software Inc." *High Go*, 8 Aug. 2019, www.highgo.ca/2019/08/08/the-difference-in-five-modes-in-the-aes-encryption-algorithm/. Accessed 1 June 2025.

Zou, Lin, et al. "Hybrid Encryption Algorithm Based on AES and RSA in File Encryption." *Lecture Notes in Electrical Engineering*, 2020, pp. 541–551, https://doi.org/10.1007/978-981-15-3250-4_68.