CRYPTOGRAPHY

Intro

Created by Alkalem, modified by Hanna3-14, modified again by Markus

28.11.2024

DISCLAIMER

Don't be discouraged if you not understanding everything (or anything)

WHAT'S CRYPTO?

WHAT'S CRYPTO?

Using math to secure communication

WHAT'S CRYPTO?

Using math to secure communication

... and more

… AND MORE

- Secure communication, E2EE Chat
- Secure storage
- Data erasure
- Verifying trustworthy origin
- Digital Rights Management (Licensing)
- Zoomer Crypto (Blockchain, NFT)
- Public Key Infrastructures (PKI)
- Signing contracts (legally binding)
- Malware: Ransomware and Obfuscation

KERCKHOFFS'S PRINCIPLES

1. The system must be practically, if not mathematically, indecipherable;
2. It should not require secrecy, and it should not be a problem if it falls into enemy hands;
3. It must be possible to communicate and remember the key without using written notes, and correspondents must be able to change or modify it at will;
4. It must be applicable to telegraph communications;
5. It must be portable, and should not require several persons to handle or operate;
6. Lastly, given the circumstances in which it is to be used, the system must be easy to use and should not be stressful to use or require its users to know and comply with a long list of rules.

La Cryptographie Militaire, 1883

KERCKHOFFS'S PRINCIPLES

1. The system must be practically, if not mathematically, indecipherable;
2. It should not require secrecy, and it should not be a problem if it falls into enemy hands;
3. It must be possible to communicate and remember the key without using written notes, and correspondents must be able to change or modify it at will;
4. It must be applicable to telegraph communications;
5. It must be portable, and should not require several persons to handle or operate;
6. Lastly, given the circumstances in which it is to be used, the system must be easy to use and should not be stressful to use or require its users to know and comply with a long list of rules.

La Cryptographie Militaire, 1883

CLASSICAL CRYPTO

CAESAR CIPHER

- Key K is a single value
- $encrypt_K(P) = (P + K)\ mod\ 26$
- $decrypt_K(C) = (C - K)\ mod\ 26$

Example for K = 23:

Plaintext:	T	H	E	Q	U	I	C	K	B	R	O	W	N	F	O	X	...
Ciphertext:	Q	E	B	N	R	F	Z	H	Y	O	L	T	K	C	L	U	...

VIGENÈRE CIPHER

- Key K is multiple letters
- Shift each letter in the plaintext according to the key letter
- Repeat K to plaintext length
- Attacks: determine key length, frequency analysis (Kasiski examination)

Plaintext:	A	T	T	A	C	K	A	T	D	A	W	N
Key:	L	E	M	O	N	L	E	M	O	N	L	E
Ciphertext:	L	X	F	O	P	V	E	F	R	N	H	R

ENIGMA

- German cipher machine during WWII
- Rotor machine with plugboard
- Used by military and government
- Broken by Polish and British cryptanalysts
- Alan Turing and the Bombe
- Rotormachines used up to the 90s (Fialka)
- Crypto AG very good company

ONE TIME PAD

- Key K is as long as the plaintext, used only once and random
- information-theoretically secure as shown by Shannon
- Key exchange needs to be done separately via a secure channel
- Mostly not realizable
- Shannon paved the way for modern cryptography
- Famous Example: Red Telephone

MODERN CRYPTO

MODERN CRYPTO

Random Number Generator	Symmetric Encryption	Asymmetric Encryption	Key-Exchange
Hashfunction	Message Authentication Code	Digital Signatures	Secret Sharing
Zero-Knowledge Proofs	Homomorphic Encryption	Multi-Party-Computation	Long-Term Encryption
Post-Quantum Crypto	Lattice-Based Crypto	Pairing-Based Crypto	Quantum Crypto

SECURITY OF CRYPTO

- Definition of security properties
- Proof that a scheme holds security guarantees
- Reduction to hard problems
- Assumptions is that the problem is actually hard
- Cryptoanalysis tried very hard to break it
- Confidentiality, Integrity, Authenticity, Non-Repudiation

SYMMETRIC ENCRYPTION

SYMMETRIC ENCRYPTION

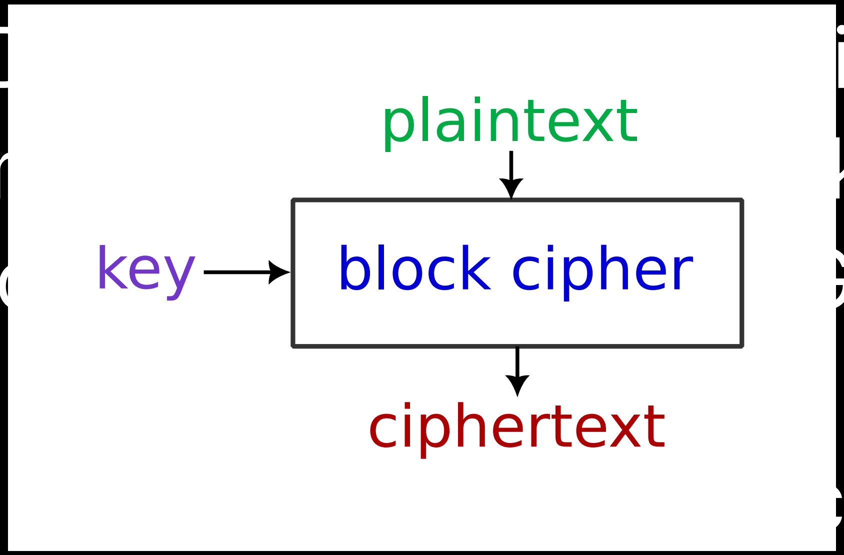
BLOCK CIPHERS

- Encrypt blocks of fixed length
- Examples: DES, IDEA, RC5, AES, Blowfish, ...
- Padding: extend messages to full block length
- Modes of operation: ECB, CBC, CTR, GCM
- Attacks:
  - against cipher: differential or linear cryptanalysis
  - different attacks against different modes of operation

SYMMETRIC ENCRYPTION

BLOCK CIPHERS

- Encrypt blocks of fixed length
- Examples: DES, ID ish, ...
- Padding: extend r k length
- Modes of operati GCM
- Attacks:
  - against cipher: ryptanalysis
  - different attacks against different modes of operation



BLOCK CIPHERS

MODES OF OPERATION

- ECB: Electronic Codebook Mode
- CBC: Cipher Block Chaining Mode
- CTR: Counter Mode
- GCM: Galois/Counter Mode

BLOCK CIPHERS

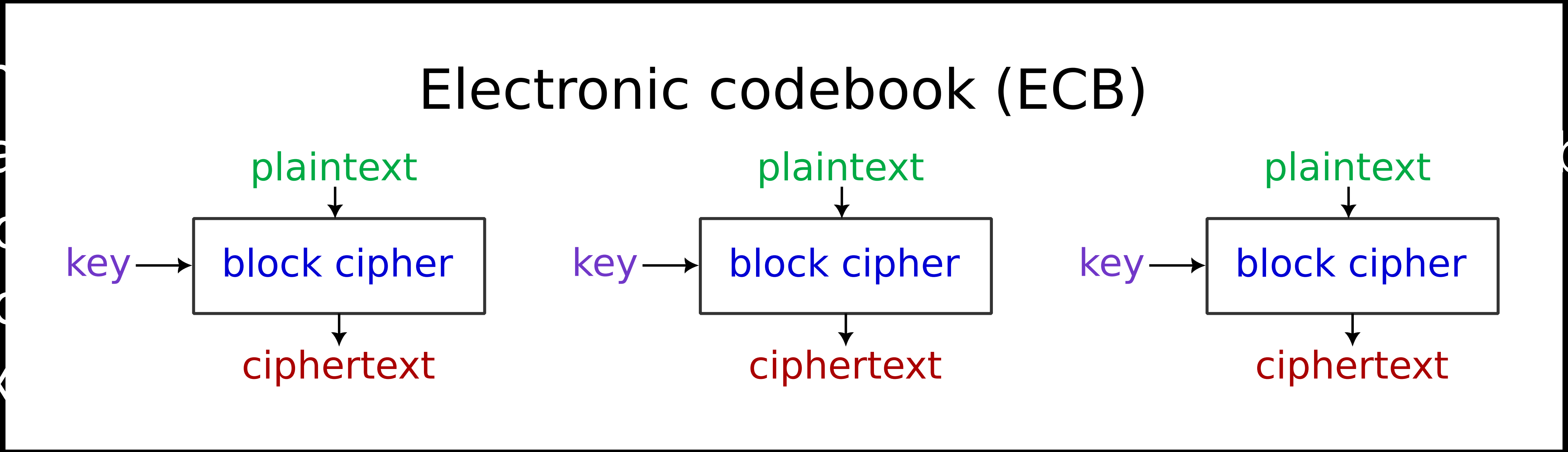
ECB: ELECTRONIC CODEBOOK MODE

- Encrypt each block separately
- Same plaintext block always encrypted to same ciphertext block
- No integrity protection
- Not secure
- Example: Tux

BLOCK CIPHERS

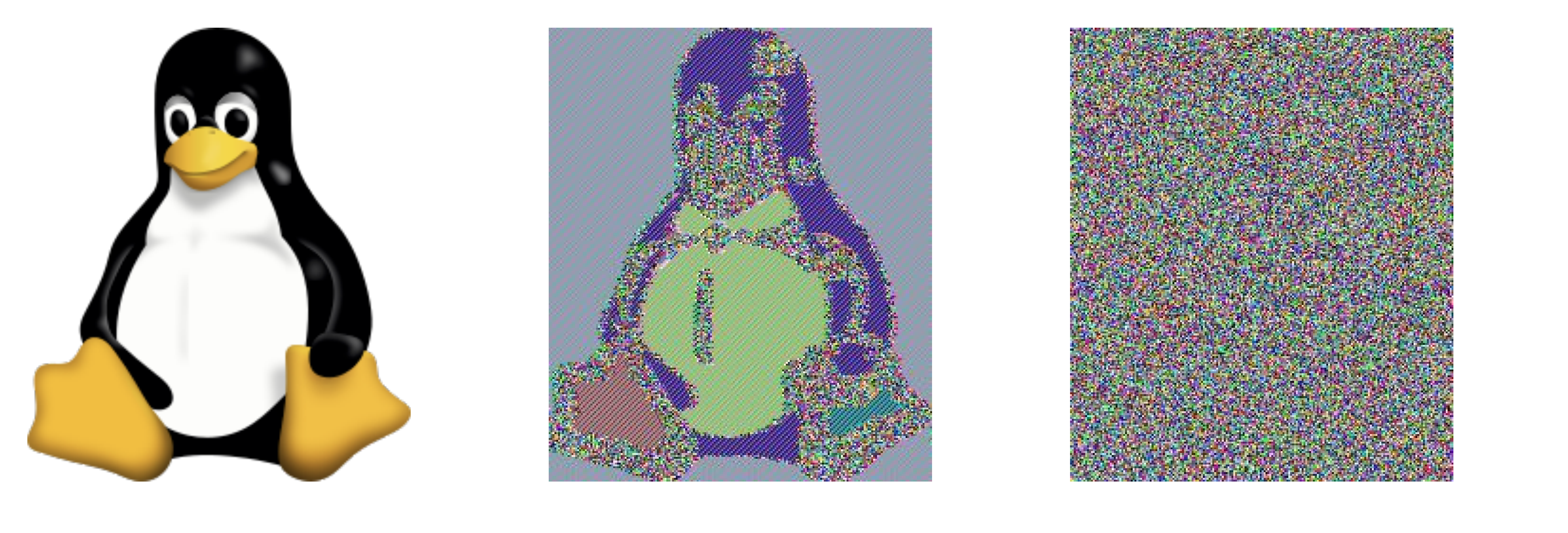
ECB: ELECTRONIC CODEBOOK MODE

- En
- Sa block
- No
- No
- Ex



BLOCK CIPHERS

ECB: ELECTRONIC CODEBOOK MODE



BLOCK CIPHERS

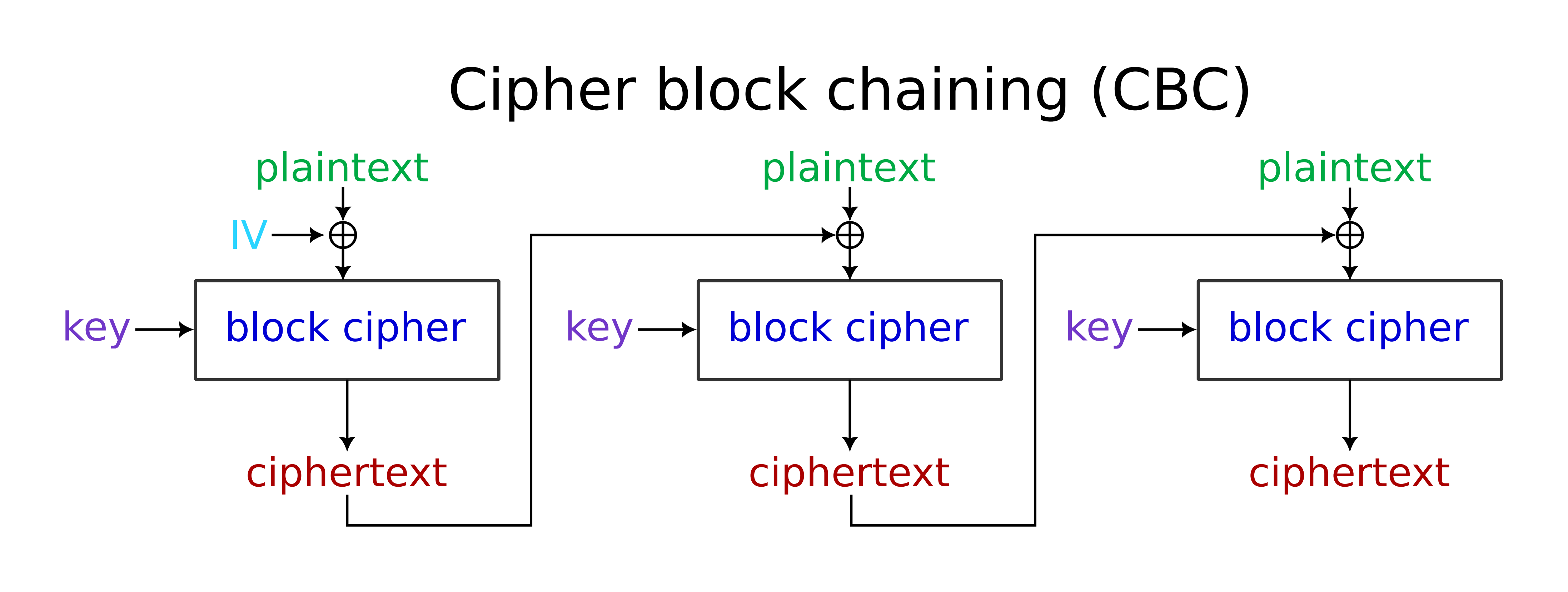
CBC: CIPHER BLOCK CHAINING MODE

- Blockchaining before it was cool
- XOR previous ciphertext block onto current plaintext before encrypting
- No integrity protection
- Use Initialization Vector for the first block

BLOCK CIPHERS

CBC: CIPHER BLOCK CHAINING MODE

- Blockc
- XOR p … ncrypting
- No inte
- Use In



HASHFUNCTION

- Fingerprint of a blob
- Produces fixed length
- Is efficient to calculate
- Security properties:
  - Should be hard to invert
  - Should be hard to find collisions
  - Should look random

HASHFUNCTION

- Fingerprint of a blob
- Produces fixed length
- Is efficient to calculate
- Security properties:
  - Should be hard to invert
  - Should be hard to find collisions
  - Should look random
- Used as a building block: MACs, Signatures

RANDOMNESS

- In most programming languages default pseudo-random number generators (PRNGs) are not cryptographically secure

  ⇒ state can be recovered

- Randomness is needed for:
  - Key generation
  - Initialization Vectors
  - Nonces
- Cryptographically secure RNGs:
  - /dev/urandom
  - Hardware RNGs
  - RNGs of the cryptographic libraries (e.g., secrets or Crypto.Random in python)

ASYMMETRIC ENCRYPTION

- Public-Key Cryptography
- Key Generation: public key and private key
- Public key can be shared, private key must be kept secret
- Examples: RSA, ElGamal, ECC

RSA

RSA

- Rivest, Shamir, Adleman, 1977
- One of the first asymmetric schemes
- Asymmetric encryption and digital signatures
- Based on the hardness of factoring large numbers

INTERLUDE: MODULAR ARITHMETIC

- $38 \equiv 14 \ (mod\ 12)$

INTERLUDE: MODULAR ARITHMETIC

- $38 \equiv 14 \ (mod\ 12)$
- Inverse: $(A * A^{-1}) \equiv 1 (mod\ C)$

INTERLUDE: MODULAR ARITHMETIC

- $38 \equiv 14 \ (mod\ 12)$
- Inverse: $(A * A^{-1}) \equiv 1(mod\ C)$
- Only the numbers that share no prime factors with C have a modular inverse $(mod\ C)$

RSA

KEY GENERATION

- Choose large prime numbers p and q
- Calculate the modulus $N = p * q$
- Calculate $\phi(N) = (p-1)*(q-1)$
- Choose e with $gcd(e,\ \phi(N)) = 1 \land 1 < e < \phi(N)$
- Calculate d as inverse of e under modulus $\phi(N)$

$$e * d \equiv 1 \mod \phi(N)$$

- Public key: $N,\ e$
- Private key: d

RSA

ENCRYPTION AND DECRYPTION

- Enc: $c = m^e \mod N$
- Dec: $c^d = (m^e)^d = m^{ed \mod \phi(N)} = m^1 \mod N$
- Textbook-RSA is homomorphic ($Enc(m_1, pk) * Enc(m_2, pk) = Enc(m_1 * m_2, pk)$) and deterministic
- Use RSA-OAEP if that is problematic

RSA

SECURITY

- Security probably based on hardness of factoring N
- Choose N large enough
- Choose e small enough, often $e = 65537$
- Use padding (e.g. RSA-OAEP)

RSA

ATTACKS

- Factoring N has complexity of about $exp(log(N)^{\frac{1}{3}}(loglogN)^{\frac{2}{3}})$, infeasible for reasonable choice of N
- in some cases, attacks in polynomial time possible:
  - Wiener's Attack: small private exponent d ($d < \frac{1}{3}N^{\frac{1}{4}}$)
  - Coppersmith's attack: for small public exponent or partially known prime factor
  - $m < N^{\frac{1}{e}}$: calculate message as root of ciphertext
  - Hastad's Broadcast Attack: Message sent to many recipients using same public exponent

DIFFIE-HELLMAN

- Key exchange protocol
- Public parameters: prime p, generator g
- Each party chooses secret a or b, computes $A = g^a \mod p$ or $B = g^b \mod p$
- Exchange A and B, compute shared secret $s = A^b \mod p = B^a \mod p = g^{ab} \mod p$
- Is a form of Multi-Party-Computation

ATTACKS ON CRYPTO

TYPICAL VULNERABILITIES

- Implementation mistake: incorrect/vulnerable custom implementation, incorporated incorrectly into application
- Conceptual mistake: incorrect use or not sufficient for use case
- Theoretic mistake: violated condition for security,
  advanced maths or theoretic computer science necessary, "read the paper"
- Well-known and documented attacks (e.g. length extension attack)
- Oracles
- Side Channels and Faults

HISTORIC FAILURES

DEBIAN OPENSSL BUG

- 2006: Debian maintainer removed random seed generation from OpenSSL
- Result: only 32,768 different keys possible
- Discovered in 2008
- Affected: Debian, Ubuntu, other distributions
- Attack: brute-force all possible keys
- CAs revoked all affected certificates and now screen for usage of these keys

PLAYSTATION 3 HOMEBREW

- 2010: At the 27C3
- Sony used ECDSA for signing games
- Fail: used the same random number for all signatures
- Result: Private key recovery
- Used to sign arbitrary code, rendering the PS3 open to homebrew

POODLE

- 2014: Padding Oracle On Downgraded Legacy Encryption
- Attack: downgrade SSL/TLS connection to SSLv3
- Exploit: padding oracle attack
- Result: decrypt SSLv3 traffic
- Fix: disable SSLv3

ROCA VULNERABILITY

- 2017: Infineon TPMs generate weak RSA keys
- Affected: TPMs, smartcards, HSMs
- Attack: factor RSA keys
- Discovered by Czech researchers
- Affected: 750,000,000 devices
- Fix: firmware update, new keys

SHA-1 COLLISION

- 2017: Google and CWI Amsterdam found practical SHA-1 collision
- attack: 2^{63} complexity
- SHA-1 deprecated since 2011
- Still used in some applications

EUCLEAK

- 2024: Side Channel Attack of Infineon Crypto Library
- Extraction of Keys from Hardware
- Power and EM Sidechannel attack
- Reversing of Infineons Library

CRYPTO TOOLS

WHERE YOU REALLY SEE THE CRYPTOGRAMS USED IN CTFS

- CyberChef
- https://factordb.com/
- OpenSSL
- GnuPG
- age / rage
- LUKS
- Sagemath (free open-source mathematics software system)
- Z3 (theorem prover)

DISCLAIMER

Don't be discouraged if you not understanding everything or anything

START PLAYING CRYPTO CTF!

- https://intro.kitctf.de/
- other platforms:
  - https://cryptohack.org/ (Easy to hard, with good explanations)
  - https://cryptopals.com/ (Implement cryptosystems and attacks)
  - https://overthewire.org/wargames/krypton (Classical crypto)
  - https://imaginaryctf.org/ (not only crypto but also other CTF challenges)