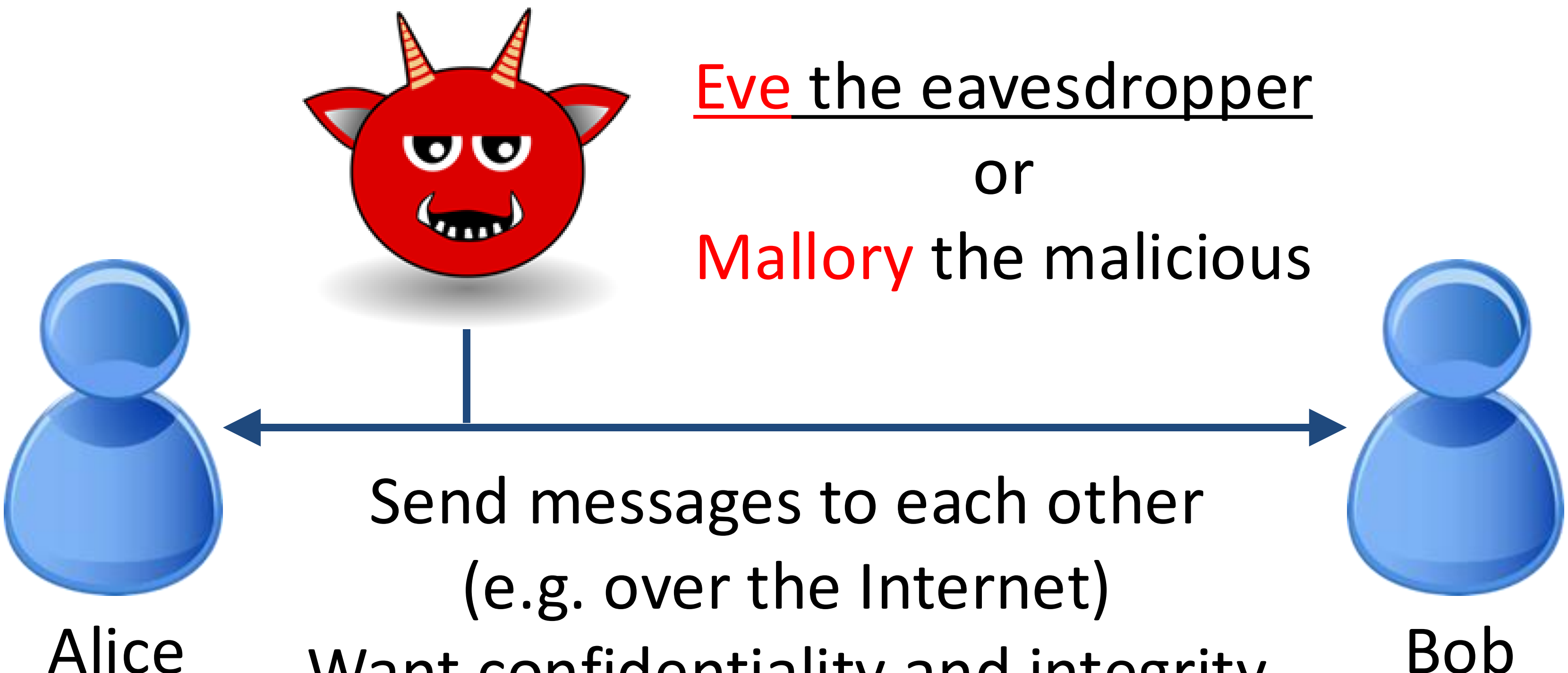
Chapter 16 – Symmetric Encryption

University of Illinois
ECE 422/CS 461

Cryptography (or Cryptology)

- Studies techniques for secure communication in the presence an adversary who has control over the communication channel



Send messages to each other
(e.g. over the Internet)
Want confidentiality and integrity

Goal of this Lecture

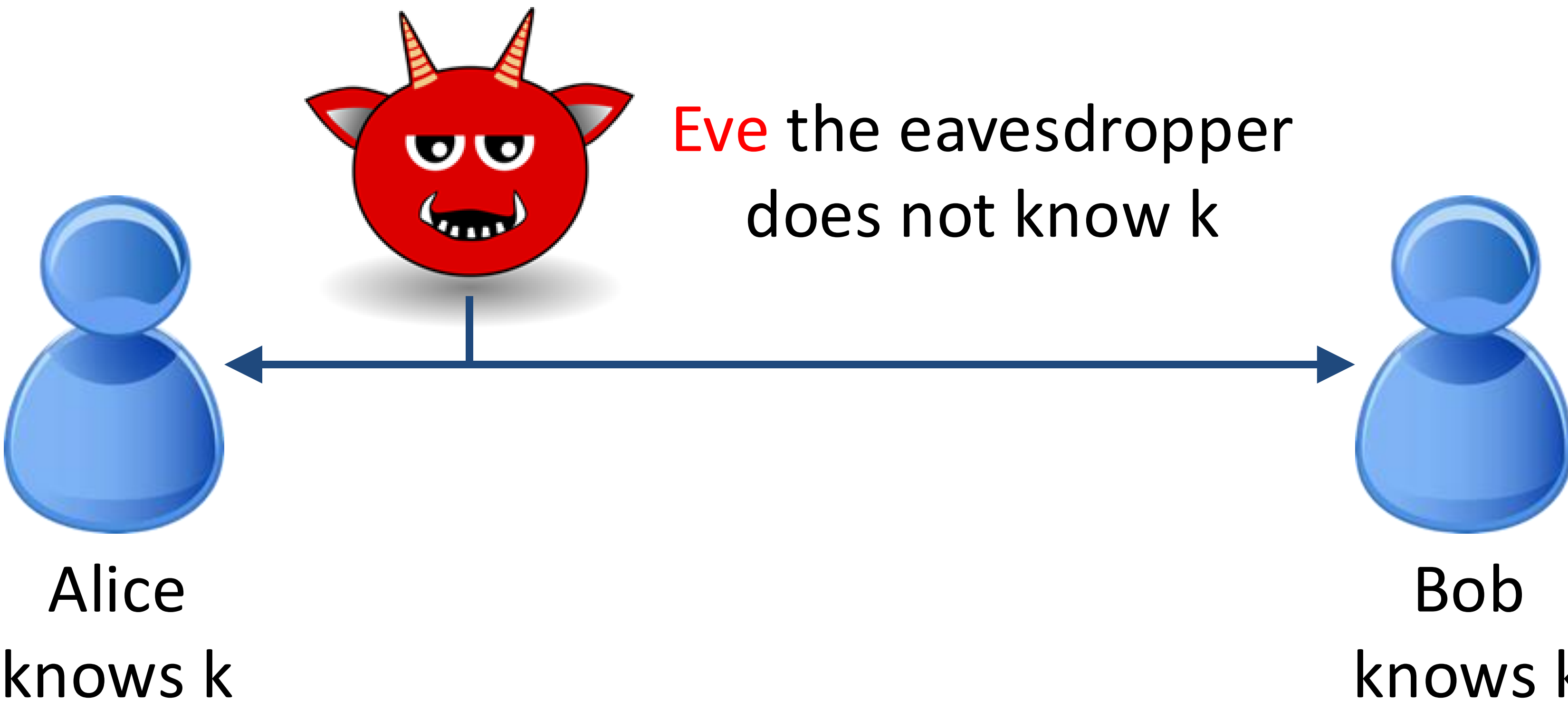
- By the end of this lecture you should know the following about symmetric encryption:
  - Interface
  - Security definition
  - Common paradigms
  - Recommended scheme and modes of operation
  - Difference between symmetric and asymmetric encryption
    - Next lecture: asymmetric encryption

Symmetric Encryption

- Allows both parties to send messages in private (an eavesdropper cannot understand)

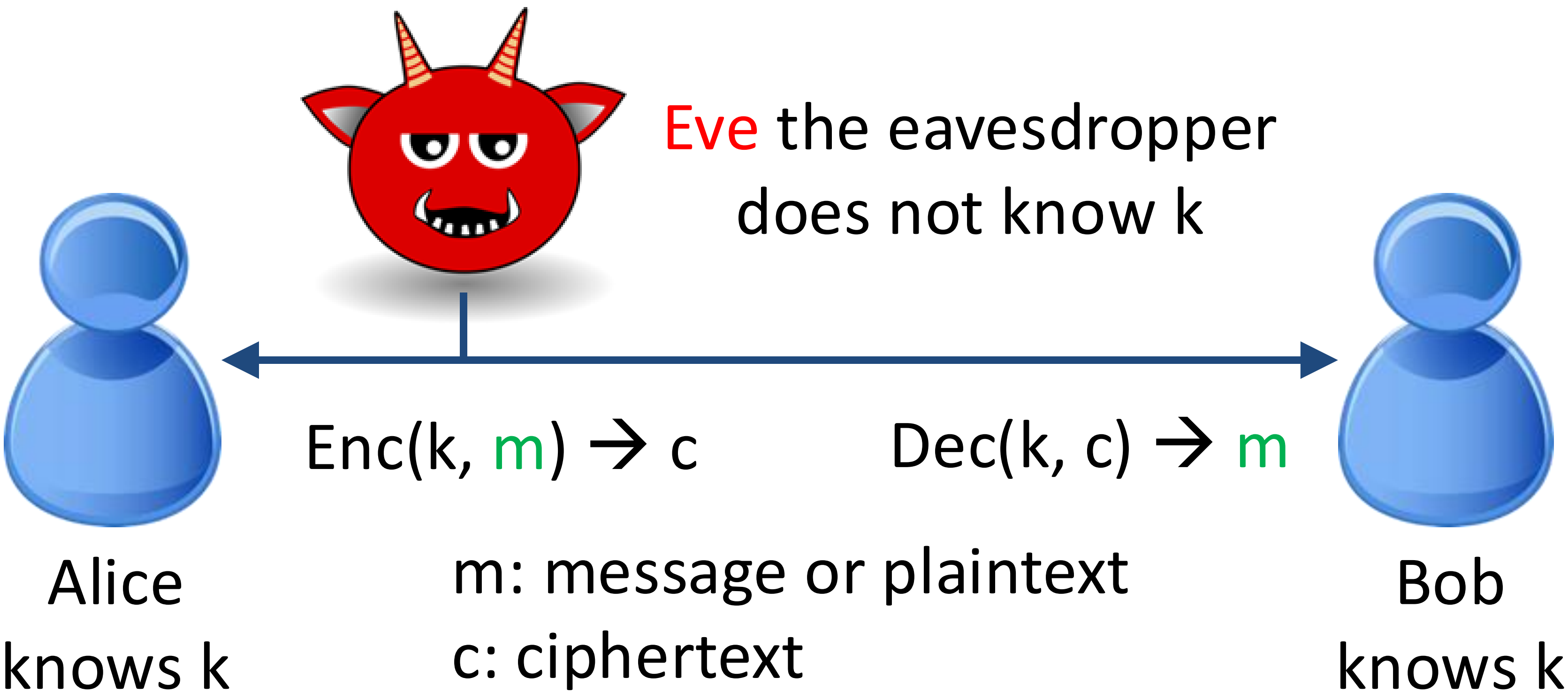
Symmetric Encryption

- Allows both parties to send messages in private (an eavesdropper cannot understand)
- Alice and Bob **must** share a secret key



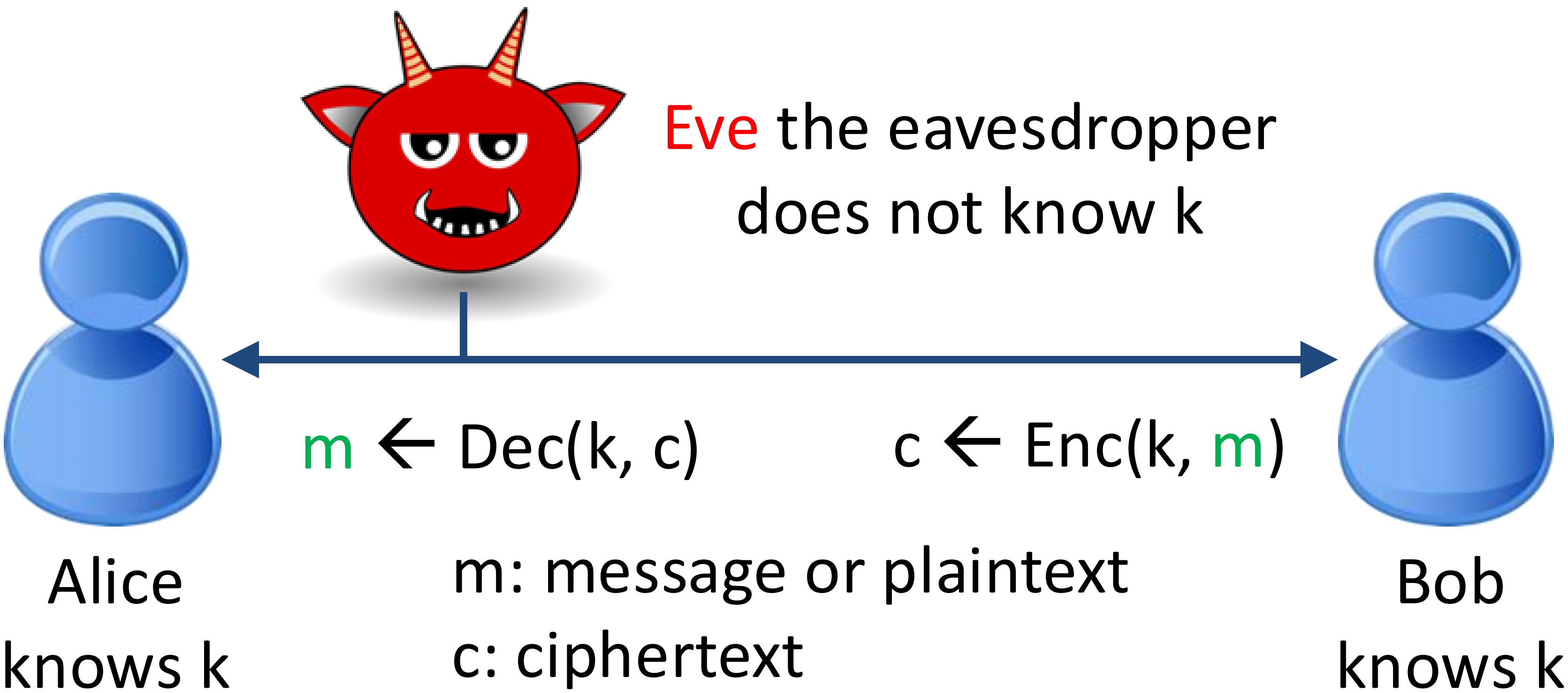
Symmetric Encryption

- Allows both parties to send messages in private (an eavesdropper cannot understand)
- Alice and Bob **must** share a secret key



Symmetric Encryption

- Allows both parties to send messages in private (an eavesdropper cannot understand)
- Alice and Bob **must** share a secret key



Outline

- Review (broken) encryption schemes in history
  - Did not think through security definition and model
- Security definition and threat model of encryption in modern cryptography

- Common paradigms & recommended schemes
- Symmetric and asymmetric encryption

Caesar Cipher

- First recorded use of encryption
  - Julius Caesar (100-44 BC)
- Replace each letter with one a fixed number of places down the alphabet
  - E.g., if secret shift (key) k = 3, then
  - Plaintext: ABCDEFGHIJKLMNOPQRSTUVWXYZ
  - Ciphertext: DEFGHIJKLMNOPQRSTUVWXYZABC
  - Plaintext: I came I saw I conquered
  - Ciphertext: L fdph L vdz L frqtxhuhg
- How to break?
  - Brute force all possible shifts

Substitution Cipher

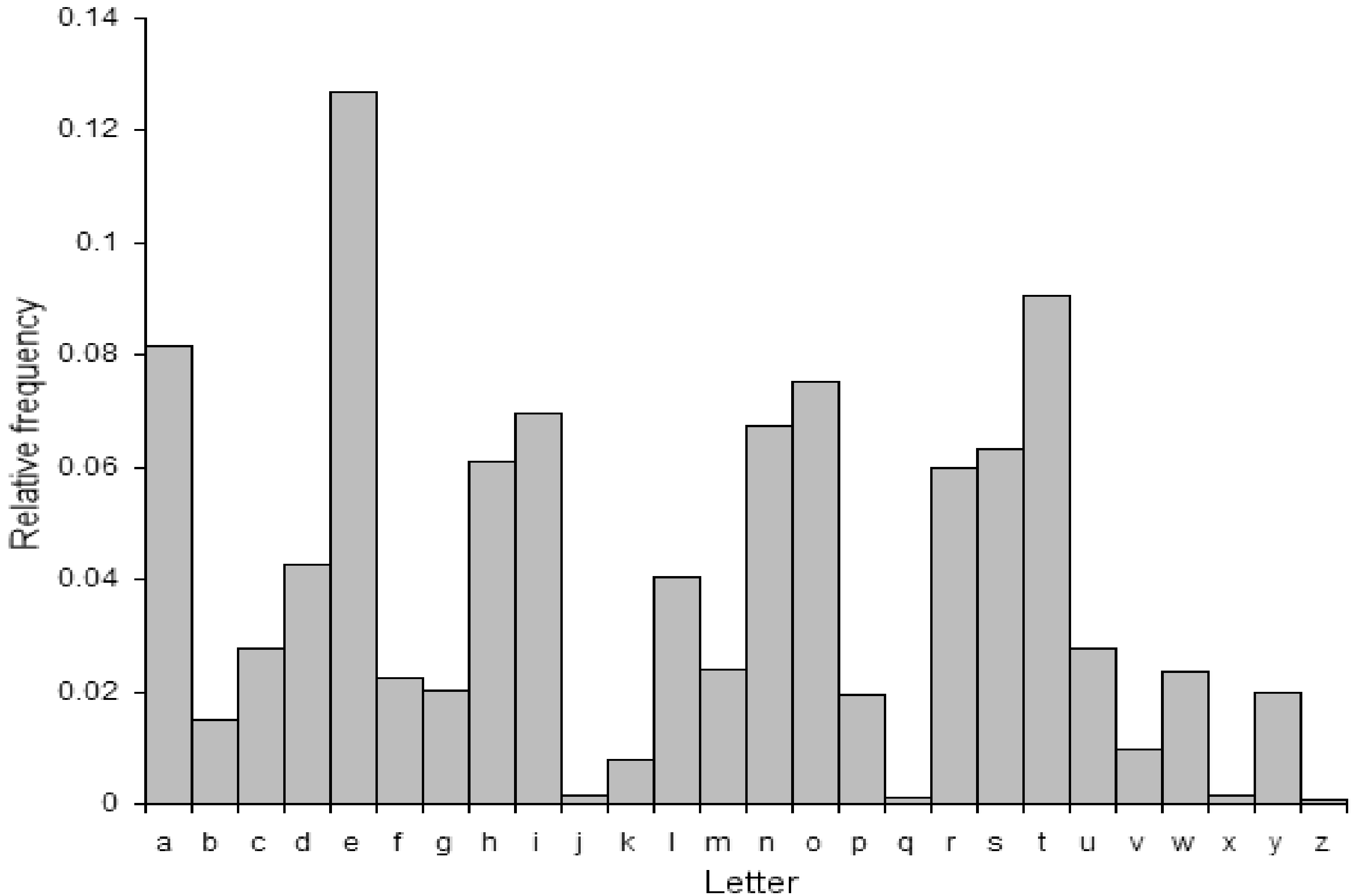
- Secret key is a secret permutation of alphabet
  - Plaintext: ABCDEFGHIJKLMNOPQRSTUVWXYZ
  - Ciphertext: GERFHSBVTKPLWUDYCXAOJNZIMQ

  - What's the secret key? How many possible keys?
  - 26! possible permutations. Can't brute-force.

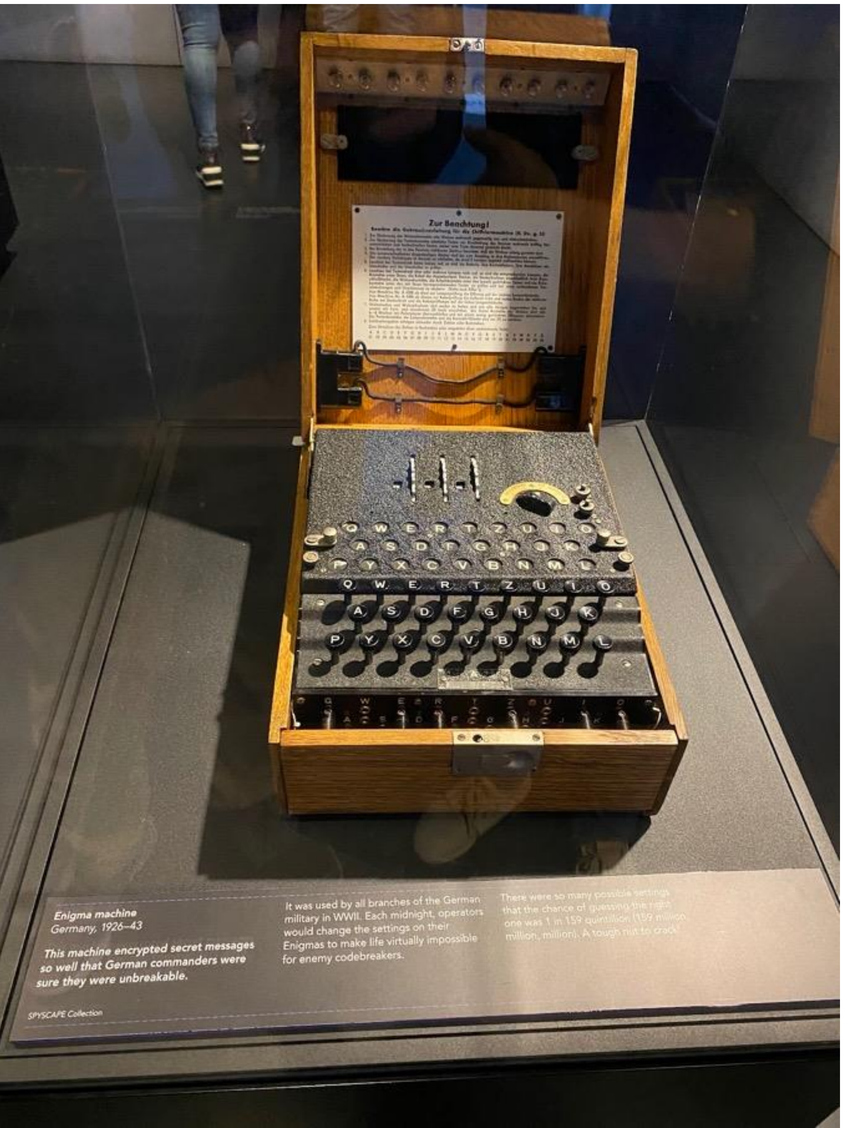
  - How to break?

Break Substitution Cipher

- English letter frequency analysis



Enigma



- A substitution cipher where the permutation table slowly changes

- Used by Germany during War World II

- Eventually broken by a British team led by Alan Turing


by hand. And so decrypt German messages.
When the Germans introduced a fourth rotor into
naval Enigma machines, engineer Doc Keen made
a new four rotor Bombe so that German messages
could keep being read. An American team also
developed a four rotor Bombe. By the end of the

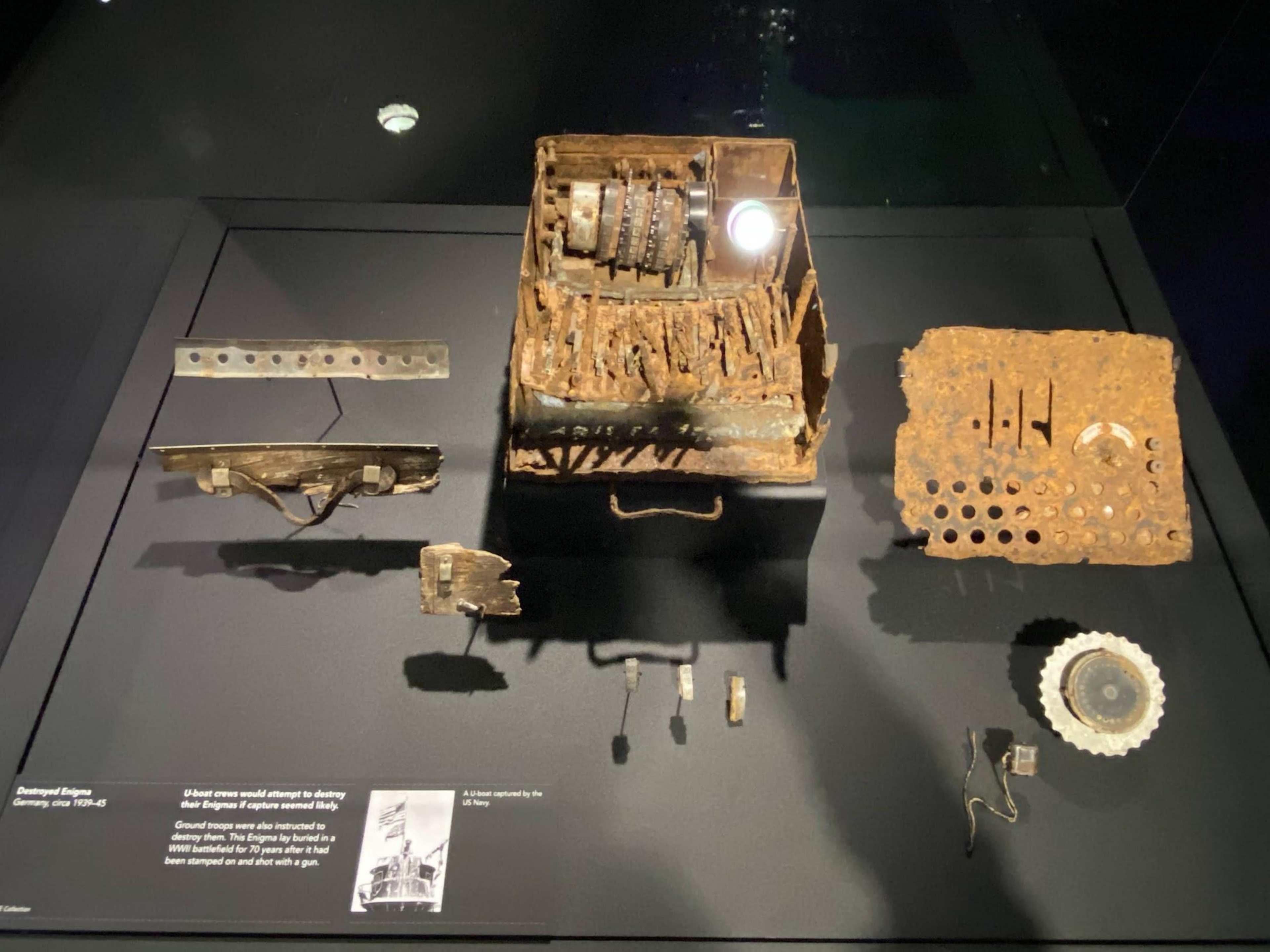
Lessons from Historic Encryption

- Need a clear and rigorous security definition that we can test a scheme against
- Try hard to test/break your encryption scheme
  - *"Any fool can create an encryption algorithm that he himself can't break",* paraphrased from Schneier

Do NOT roll your own crypto!

Lessons from Historic Encryption

- Need a clear and rigorous security definition that we can test a scheme against
- Try hard to test/break your encryption scheme
- Do NOT roll your own crypto!
- Assume secret keys and **public** algorithm
  - Kerckhoffs's Principles (1883): *It should not require secrecy [except for key], and it should not be a problem if it* [the algorithm] *falls into enemy hands*.
  - Claude Shannon (1949): *the enemy knows the system*


Destroyed Enigma
Germany, circa 1939–45
U-boat crews would attempt to destroy their Enigmas if capture seemed likely.
Ground troops were also instructed to destroy them. This Enigma lay buried in a WWII battlefield for 70 years after it had been stamped on and shot with a gun.
A U-boat captured by the US Navy.

Lessons from Classic Encryption

- Need a clear and rigorous security definition that we can test a scheme against
- Try hard to test/break your encryption scheme
- Do NOT roll your own crypto!
- Assume secret keys and **public** algorithm
- Assume eavesdropper obtains some (or many) plaintext-ciphertext pairs (of its choosing!)
  - If we consider a single letter, even Caesar cipher and substitution cipher are secure!

Modern Definition of Encryption

- A game with attacker Eve:
  - We (proponent of a cipher) pick a random key k

Encryption does not hide message length

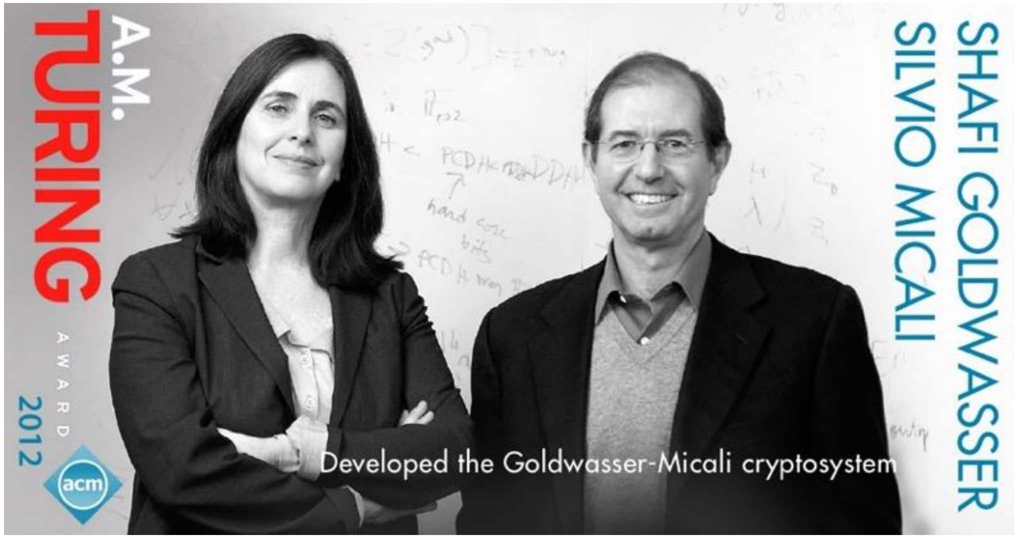
  - Eve picks two messages m_0 and m_1 of equal length
  - We flip a coin b ← {0, 1} and give Eve Enc(k, m_b)
  - Eve guesses b. Encryption is insecure if Eve wins with > 0.5 + ε probability; secure if ≈ 0.5 probability

Modern Definition of Encryption

- A game with attacker Eve:
  - We (proponent of a cipher) pick a random key k
  - Eve can ask for encryptions of any messages
    - I.e., pick any m and get back Enc(k, m), and repeat any (feasible) number of times
  - Eve picks two messages m_0 and m_1 of equal length
  - We flip a coin b $\leftarrow$ {0, 1} and give Eve Enc(k, m_b)
  - Eve can ask for encryptions of any messages
  - Eve guesses b. Encryption is insecure if Eve wins with > 0.5 + ε probability; secure if ≈ 0.5 probability

Indistinguishability under Chosen Plaintext Attacks (IND-CPA)

- A game with attacker Eve:
  - We (proponent of a cipher) pick a random key k
  - Eve can ask for encryptions of any messages
  - Eve picks two messages m_0 and m_1 of equal length
  - We flip a coin b ← {0, 1} and give Eve Enc(k, m_b)
  - Eve can ask for encryptions of any messages
  - Eve guesses b. Secure iff Eve wins with ≈ 0.5 probability.



Is IND-CPA Too Stringent?

- A game with attacker Eve:
  - We (proponent of a cipher) pick a random key k
  - Eve can ask for encryptions of any messages
  - Eve picks two messages m_0 and m_1 of equal length
  - We flip a coin $b \leftarrow \{0, 1\}$ and give Eve $\mathrm{Enc}(k, m_b)$
  - Eve can ask for encryptions of any messages
  - Eve guesses b. Secure iff Eve wins with ≈ 0.5 probability.

- What if Eve asks for $\mathrm{Enc}(k, m_0)$ and $\mathrm{Enc}(k, m_1)$ and compares with $\mathrm{Enc}(k, m_b)$? An easy win?

Is IND-CPA Too Stringent?

- What if Eve asks for Enc(k, m_0) and Enc(k, m_1) and compares with Enc(k, m_b)? An easy win?

- No, IND-CPA is achievable!

- Need **randomized** encryption
  - Encryption of the same message (under the same key) must change every time!

One-Time Pad

- Perfect (but impractical) encryption [Shannon 1949]
- The secret **key** shared by Alice and Bob is an infinitely long random binary string, called **pad**
- Plaintexts and ciphertexts are also binary strings
- Enc/Dec work by XORing with pad bit by bit
  - c = Enc(k, m): $c[i] = pad[i] \oplus m[i]$
  - m = Dec(k, c): $m[i] = pad[i] \oplus c[i]$
- One-time: never reuse portions of pad
- Perfect secrecy (ε=0 in IND-CPA). Why?

IND-CPA for One-Time Pad

- A game with attacker Eve:
  - We (proponent of a cipher) pick a random key (pad)
  - Eve can ask for (any number of) pad ⊕ m
  - Eve picks two messages m_0 and m_1 of equal length
  - We flip a coin b ← {0, 1} and give Eve pad ⊕ m_b
  - Eve can ask for (any number of) pad ⊕ m
  - Eve guesses b. Secure iff Eve wins with ≈ 0.5 probability
- Again, crucial to never reuse pad!
- Every ciphertext bit c[i] is 0 or 1 with 50/50 chance **completely independent of the value of m[i]**

One-Time Pad

- Impractical: need to share unrealistically long keys (pad)

- But we can borrow its principle

Stream Cipher

- To obtain a practical encryption scheme, we just need to generate a long & random pad
  - Alice and Bob share k. Both compute

    $$\text{pad} = H(k||1) \; || \; H(k||2) \; || \; H(k||3) \; || \; \ldots$$

  - If H is **pseudorandom**, infeasible to distinguish pad from a truly random one-time pad

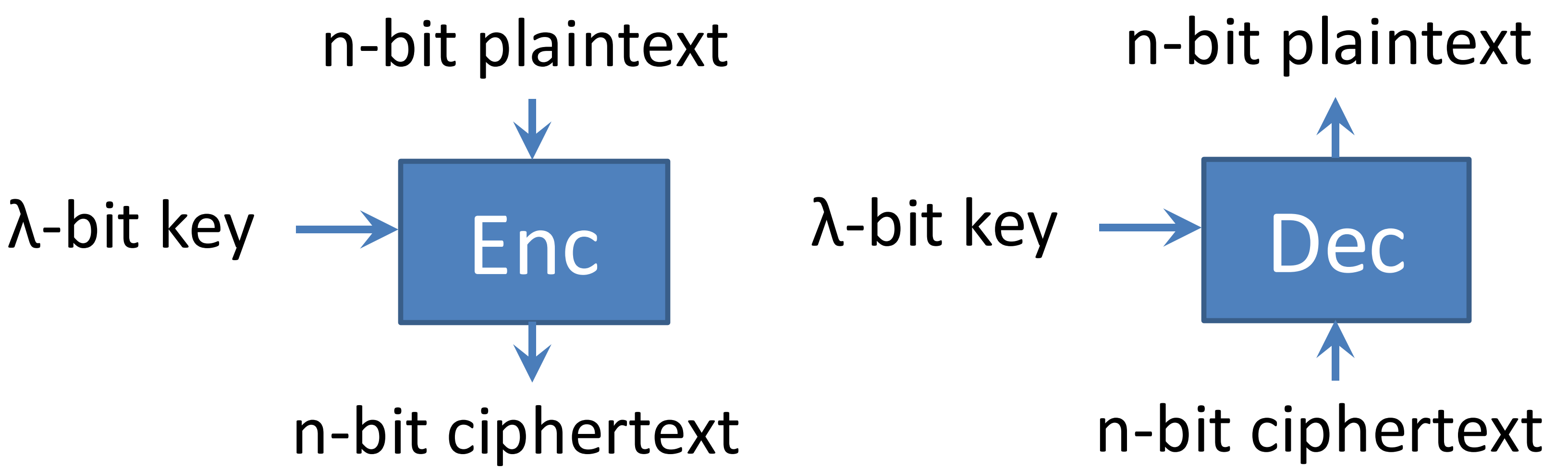
  - $c[i] = pad[i] \oplus m[i] \qquad m[i] = pad[i] \oplus c[i]$

Stream Cipher

- Using a pseudorandom crypto hash function as one-time pad is fine but not best option
  - Cryptoanalysis of hash functions focus on collision resistance rather than pseudorandomness
  - Also an overkill, no need for arbitrarily long input
- The “right” primitive is pseudorandom function (PRF), which takes fixed-length inputs
- Stream Cipher is perfectly reasonable. Has some use (e.g., Salsa20), though not mainstream.

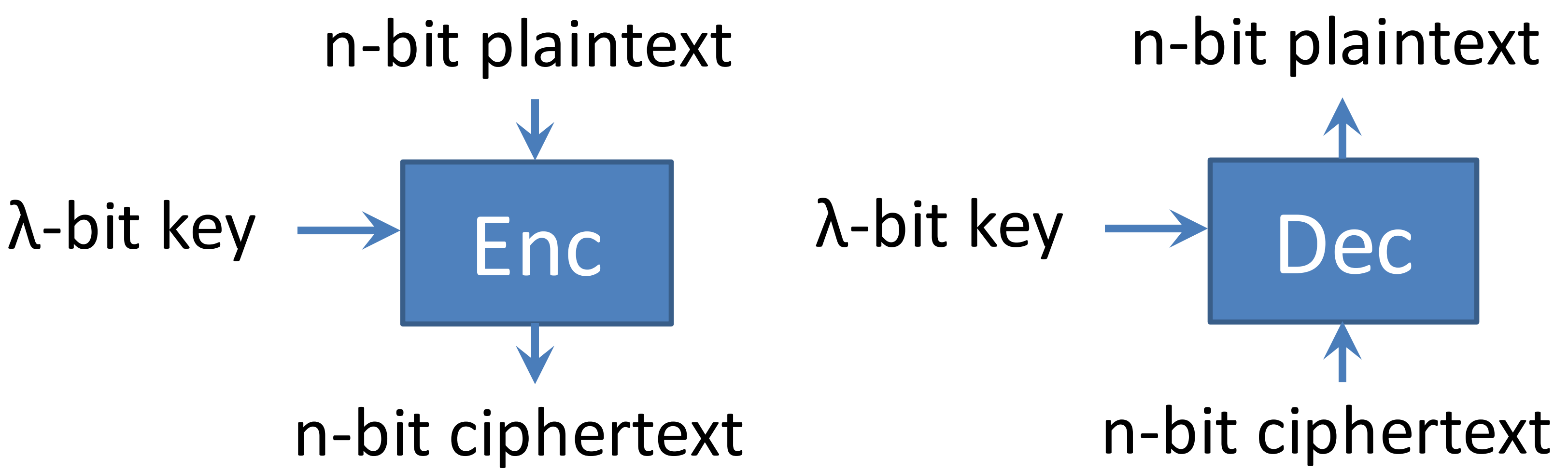
Current Mainstream: Block Cipher

- Under a fixed key k, 1-to-1 mapping between the 2^n plaintexts and 2^n ciphertexts
  – Enc and Dec are inverse permutations of each other
- Without knowing k, infeasible to distinguish from a truly random permutation



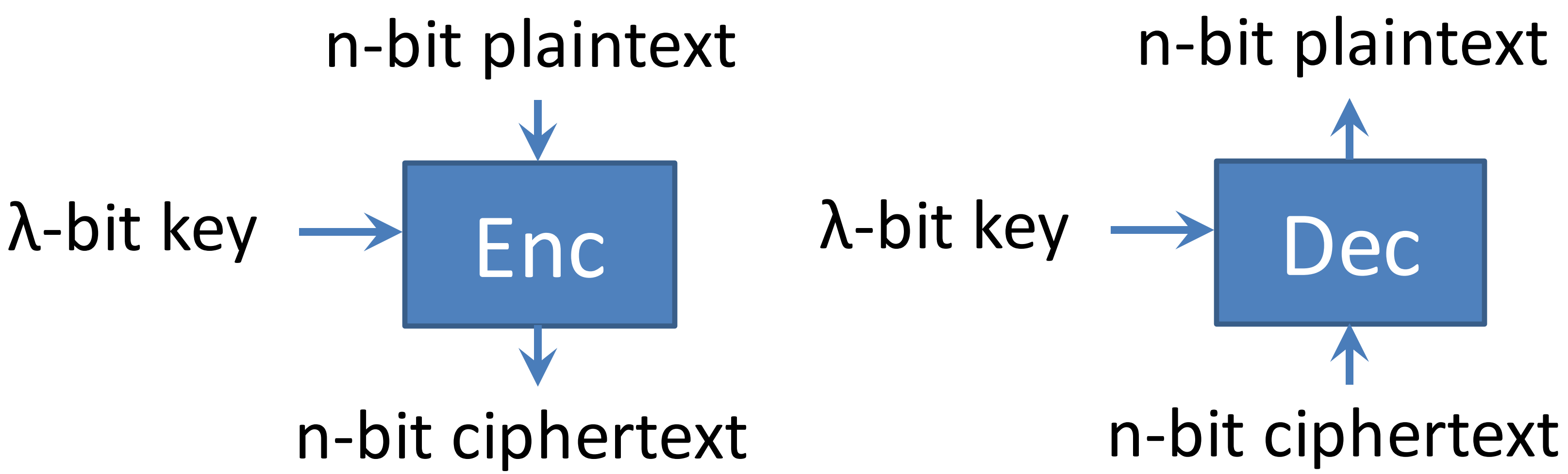
Current Mainstream: Block Cipher

- Isn't this just a substitution cipher that operates on a much larger (2^n) alphabet?
- Isn't it deterministic? Didn't we say substitution cipher and deterministic cipher are insecure?
- Yes, yes and yes … Will address these soon.



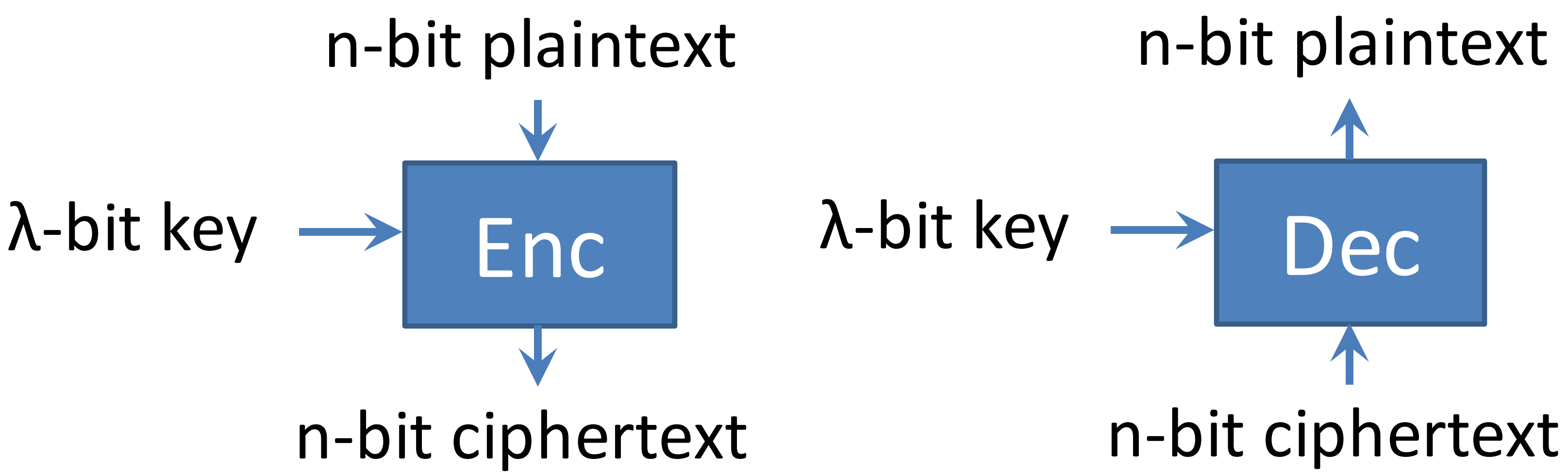
Block Cipher

- ~~DES (Data Encryption Standard)~~
  - Standardized by FIPS in 1976
  - Key size λ = 56, block size n = 64
    - 2^{56} was reasonable security back then but too weak now
  - Weakened in 1992, broken in 1997



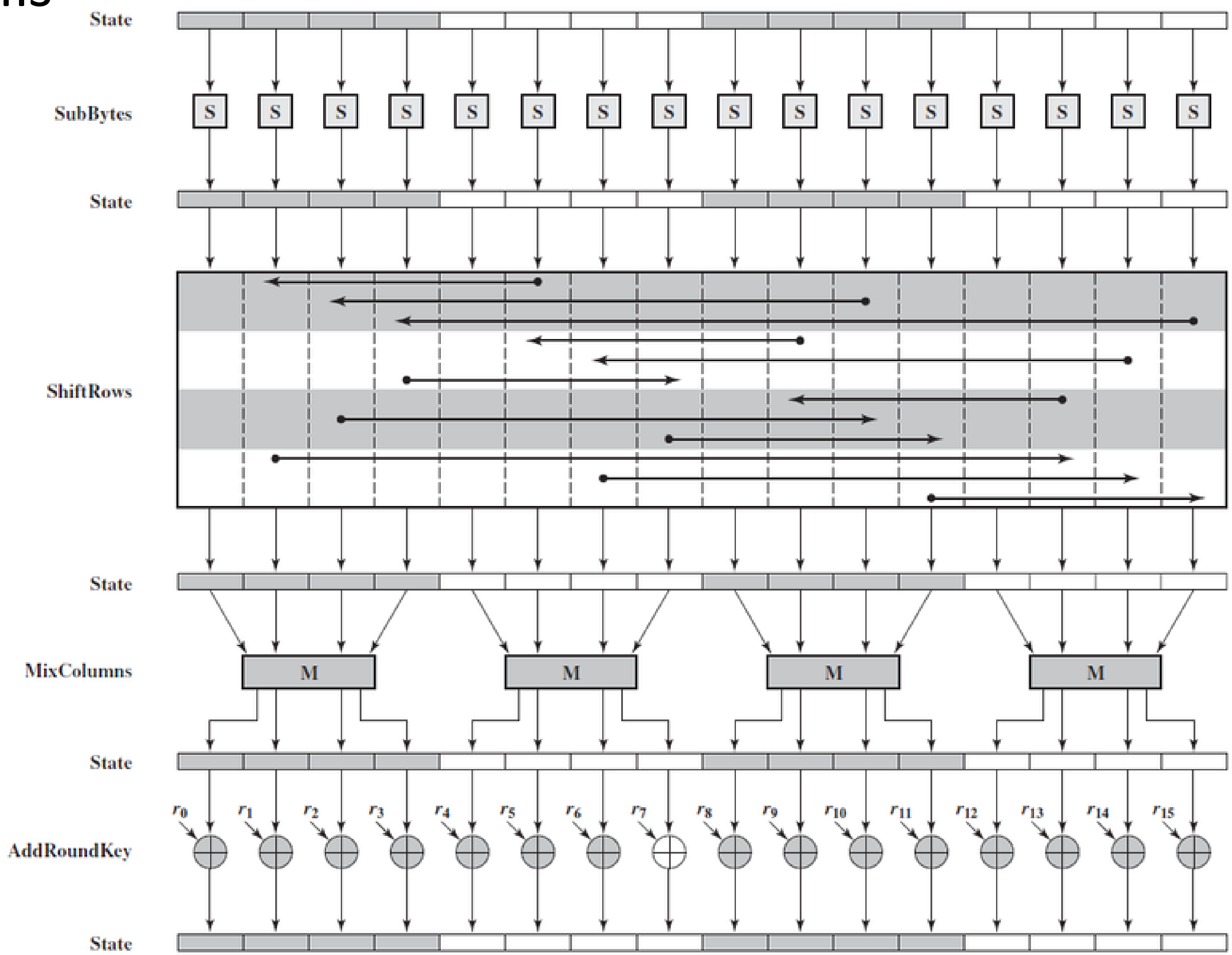
Block Cipher

- AES (Advanced Encryption Standard)
  - Standardized by NIST in 2001
  - Block size n = 128, key size λ = 128, 192, or 256
    - Correspond to AES-128, AES-192, AES-256



AES-128

Ten rounds of this

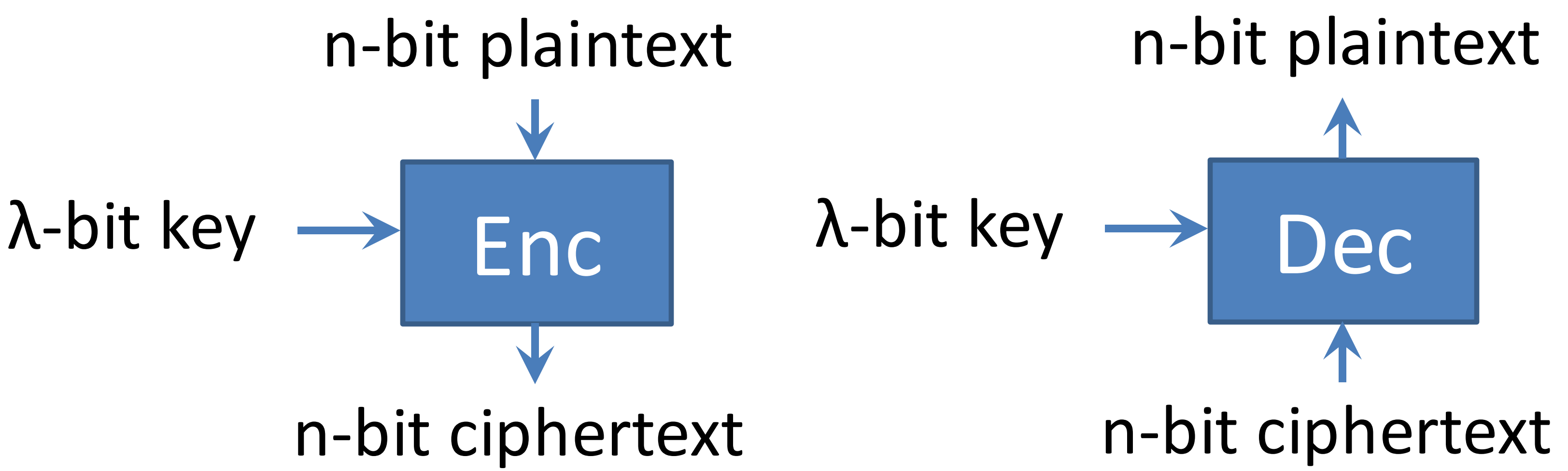


Block Cipher

- AES (Advanced Encryption Standard):
  - Standardized by NIST in 2001
  - Block size n = 128, key size λ = 128, 192, or 256
    - Correspond to AES-128, AES-192, AES-256
  - Slightly weakened in 2011 ($2^{126.1}$, $2^{189.7}$, $2^{254.4}$)
  - Still the leading and recommended scheme today
    - Why is AES not replaced despite weakened?
    - Possibly because a block cipher is harder to design (need to be invertible with the key)

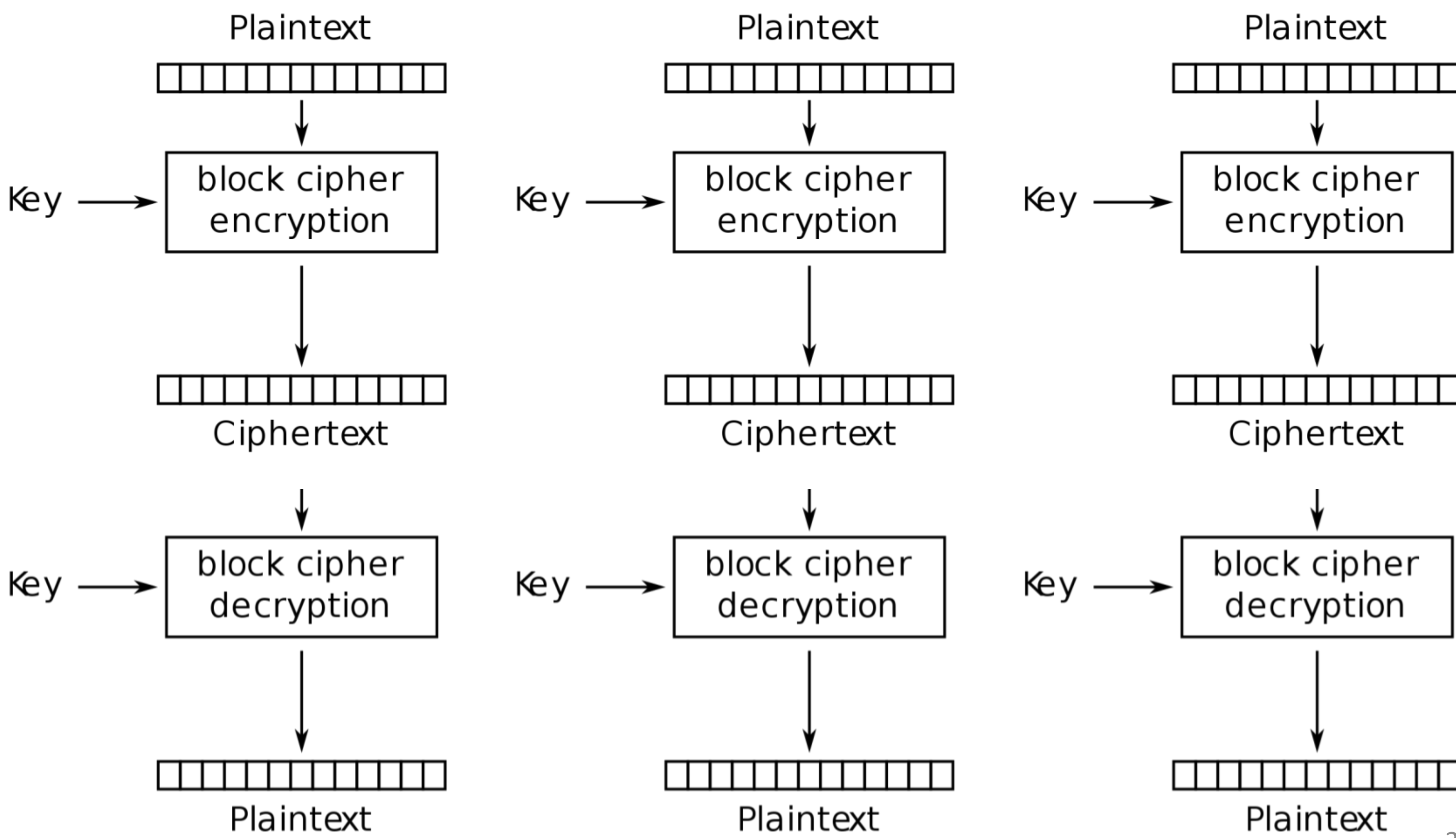
Block Cipher

- Why use block cipher over stream cipher?
  - One bad reason: block cipher's abstraction matches layman intuition for encryption
  - Better reason: more use → better studied → more likely to be secure → more use



Block Cipher Modes: ECB

- Electronic codebook = as a substitution cipher



IND-CPA

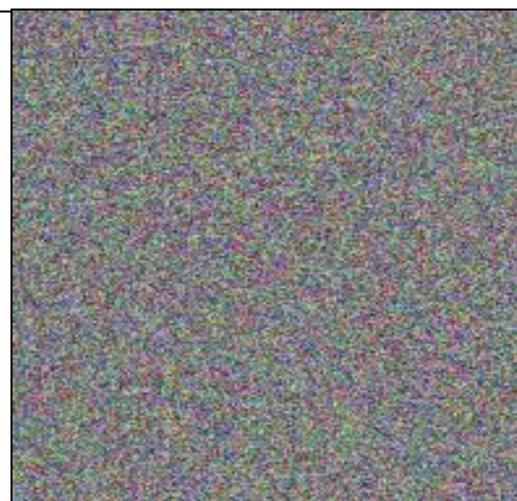
- A game with attacker Eve:
  - We (proponent of a cipher) pick a random key k
  - Eve can ask for encryptions of any messages
  - Eve picks two messages m_0 and m_1 of equal length
  - We flip a coin $b \leftarrow \{0, 1\}$ and give Eve $Enc(k, m_b)$
  - Eve can ask for encryptions of any messages
  - Eve guesses b. Insecure if Eve wins with $0.5 + \varepsilon$ probability.

- Eve asks for $Enc(k, m_0)$ and $Enc(k, m_1)$ and compares with $Enc(k, m_b)$. An easy win!

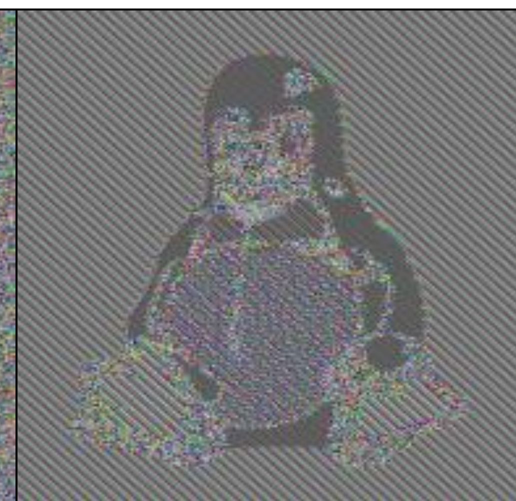
Electronic Codebook (ECB) Mode

- **Avoid!!!**
  - Unfortunately, the default mode of most libraries
- Deterministic encryption, not IND-CPA secure

Plaintext

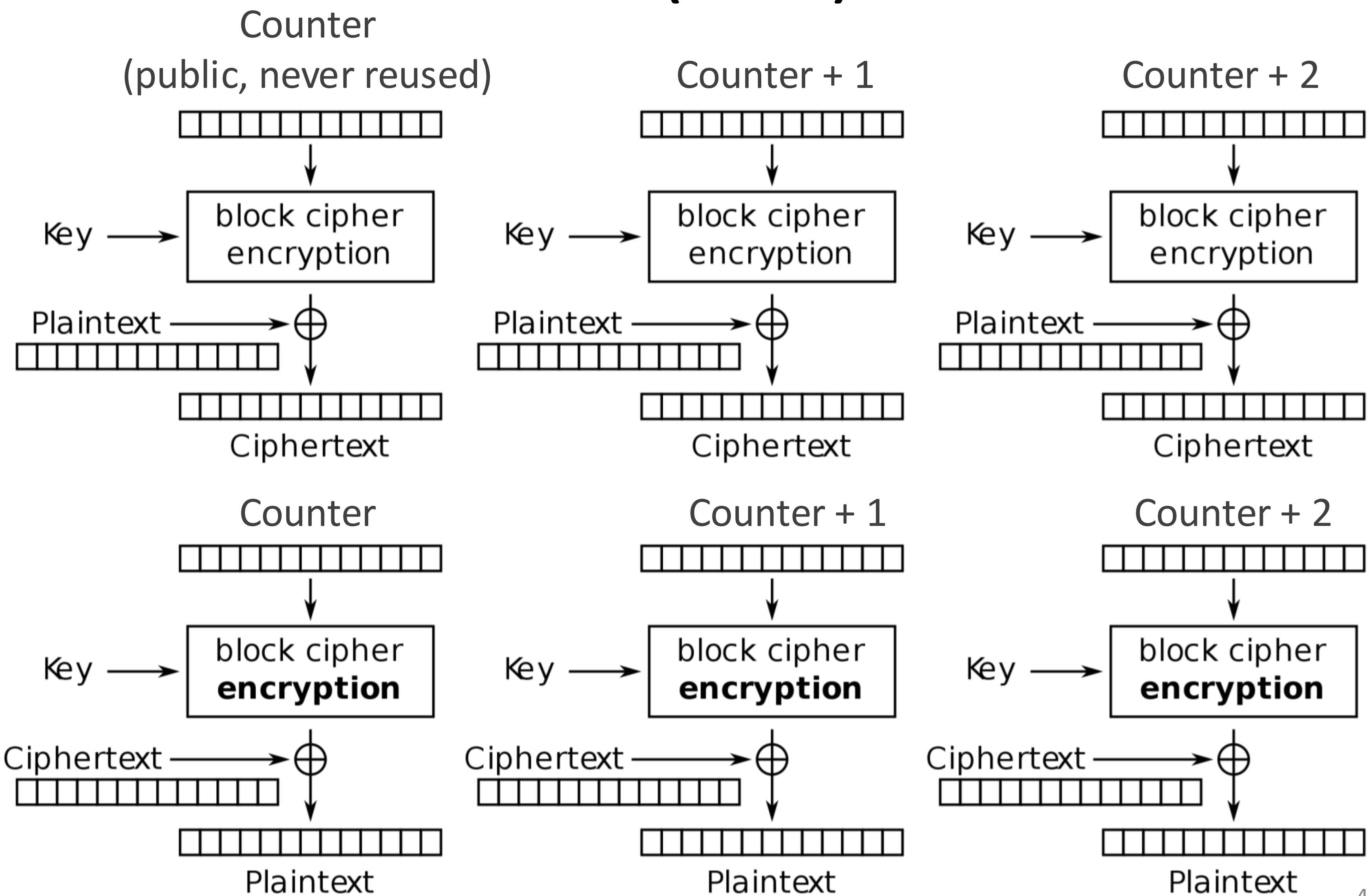


randomized encryption

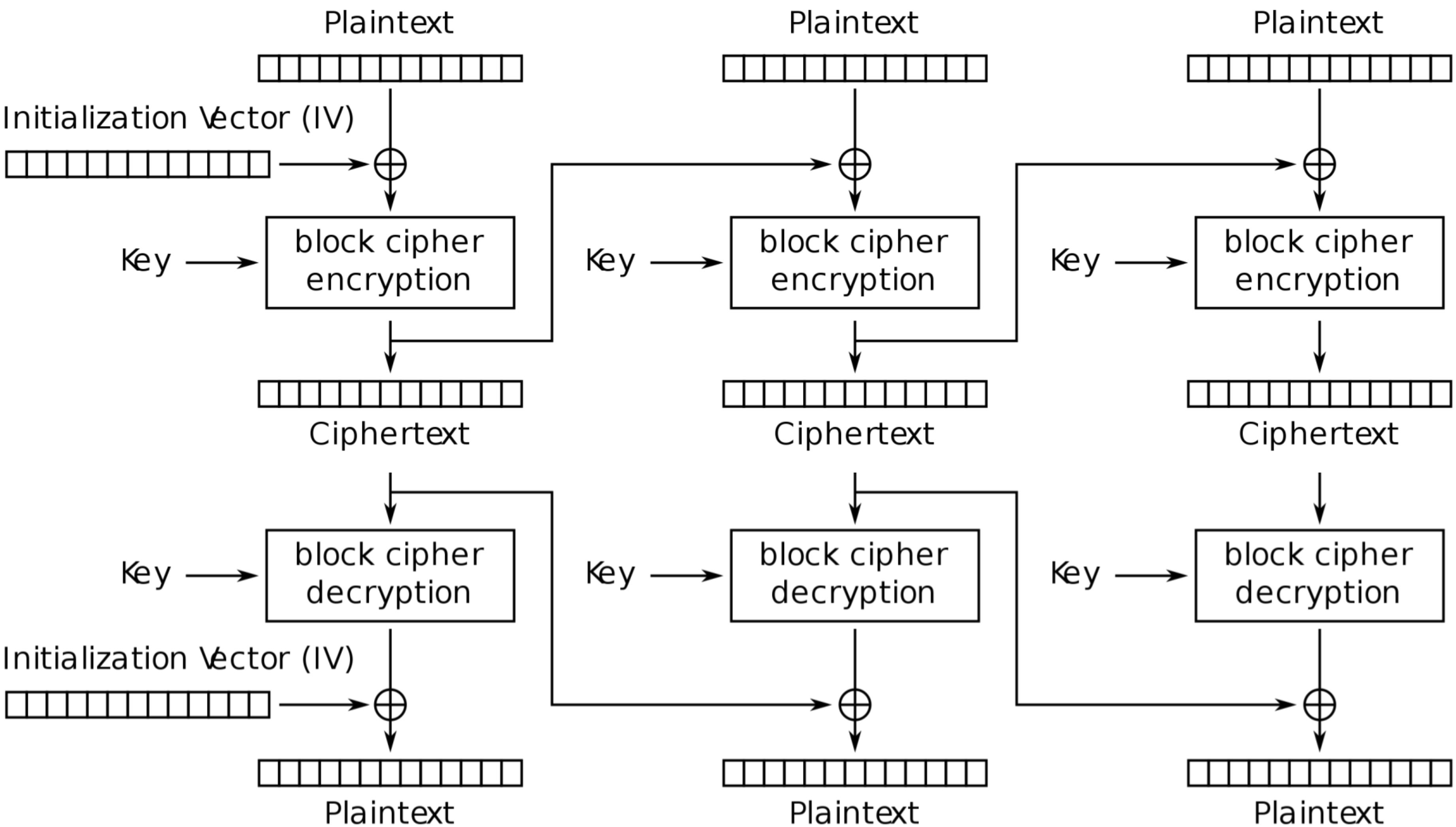


ECB mode

Counter (CTR) Mode



Cipher Block Chaining (CBC) Mode



Block Cipher Modes Summary

- ECB: as substitution cipher. Avoid!

- CTR: as stream cipher and one-time pad
- CBC: add more dependency among blocks
- Other less common modes: CFB, OFB, …

- Some modes also provide integrity, e.g., GCM
  – A legit argument for block cipher over stream cipher (which needs orthogonal mechanisms for integrity)

Symmetric Encryption Summary

- Shared key k, c = Enc(k, m), m = Dec(k, c)
- Security definitions: allow Eve to get many plaintext-ciphertext pairs (e.g., IND-CPA)
- Must use randomized encryption
- Paradigms: stream cipher and block cipher
- Currently mainstream: AES, but avoid ECB mode