CIS 4360
Secure Computer Systems

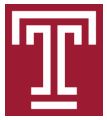
Symmetric Cryptography

Professor Qiang Zeng

Spring 2017

Previous Class

- Classical Cryptography
  - Frequency analysis
  - Never use home-made cryptography
- Goals of Cryptography
  - Confidentiality, data integrity, authentication, non-repudiation
- Building blocks of Cryptography
  - Cryptographic hash, encryption, MAC, digital signature
- Cryptographic Hash
  - Password storage, verifying data integrity
  - Do not use MD5 and SHA-1

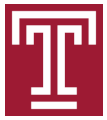


Previous class…

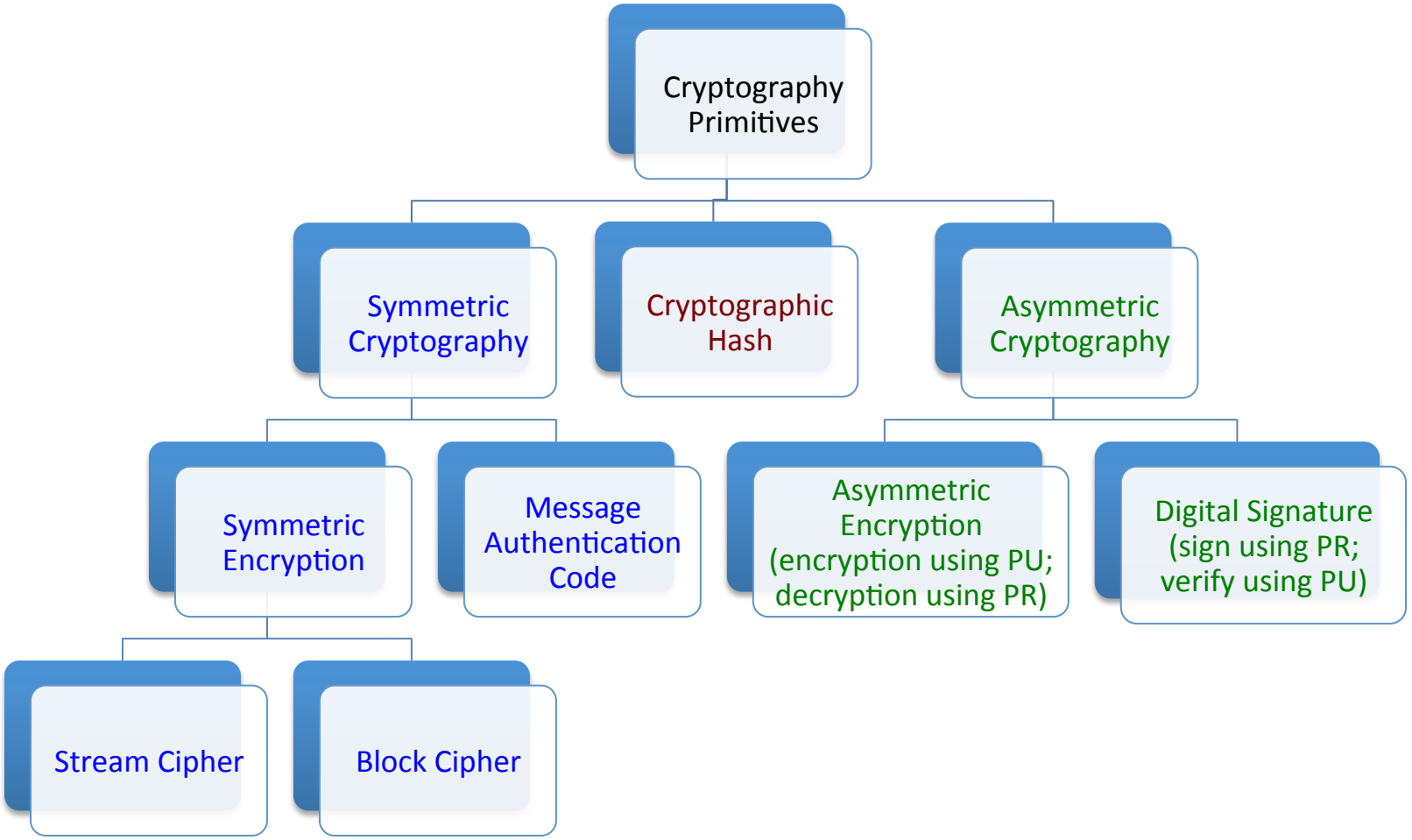
By sending m and Hash(m), can the receiver verify the integrity of m, i.e., to verify whether m has been manipulated during the transmission? If not, what should you do?

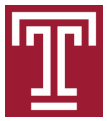
No, cryptographic hash can be used to verify data integrity only if the integrity of the hash value itself is assured. Typically, if the message and the digest value are transmitted in the same channel, it is in vain as the adversary can replace both the message and the corresponding hash value.

Message Authentication Code and Digital Signature are used when the hash value can also be attacked (discussed later)



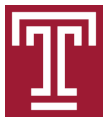
Cryptography Primitives





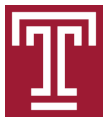
Notation

- C: ciphertext
- P: plaintext
- K: key
- PR: private key
- PU: public key
- E: encryption; e.g., C = E (K, P)
- D: decryption; e.g., P = D(K, C)
- H: hash
- ||: e.g., x || y means x concatenated with y

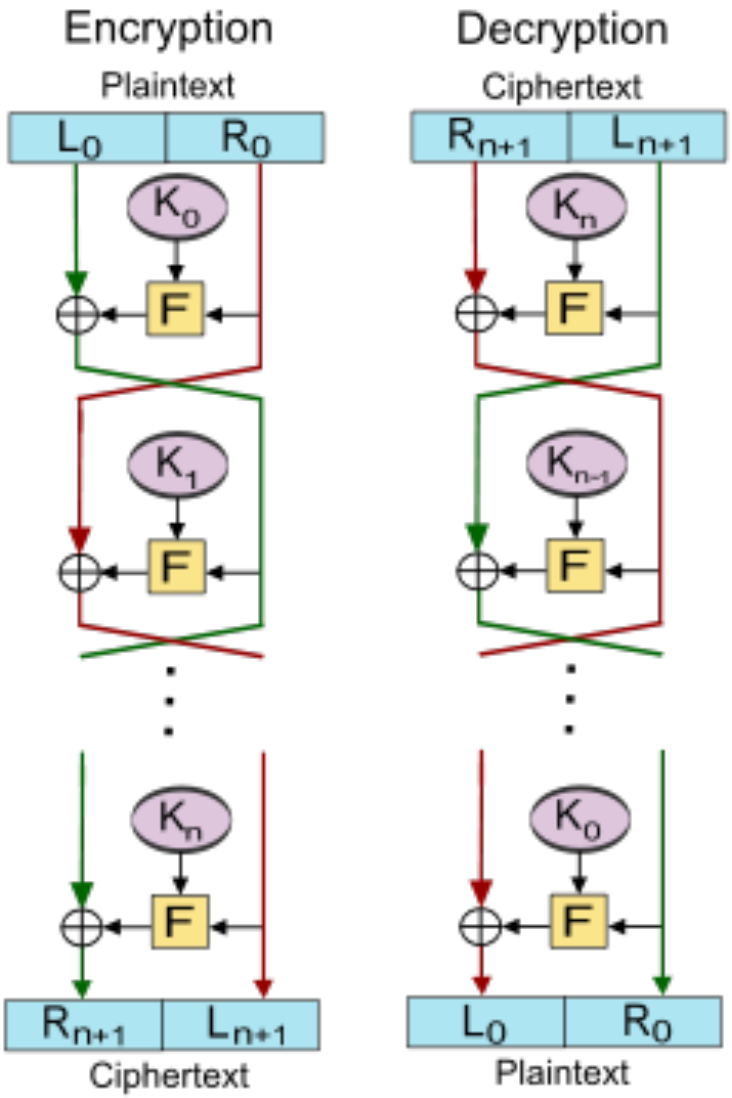


Symmetric Block Cipher - DES

- DES: Data Encryption Standard
- NIST symmetric encryption standard 1976-2001
- Already broken; 56-bit key size is too small
- Block size: 64 bits
  - 64-bit plaintext input; 64-bit ciphertext output
- How it works?
  - Feistel structure: input block is divided into halves and processed alternatively
  - XOR, Substitution (using the S-box), and Permutation (using the P-box)

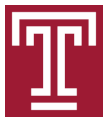


Feistel Structure



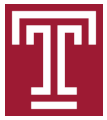
Symmetric Block Cipher – Triple-DES

- Triple-DES: still widely used
  - $C_1 = E(K_1, P)$, $C_2 = D(K_2, C_1)$, $C_3 = E(K_3, C_2)$
  - Key size: 3 x 56 = 168 bits; secure
- When k_1 != k_2 and $k_1 = k_3$
  - Key size: 2 x 56 = 112 bits; insecure
- When $k_1 = k_2 = k_3$, 3-DES becomes DES, since the second operation (D) offsets the first (E)
  - It provides compatibility with DES when needed
- Disadvantage: slow



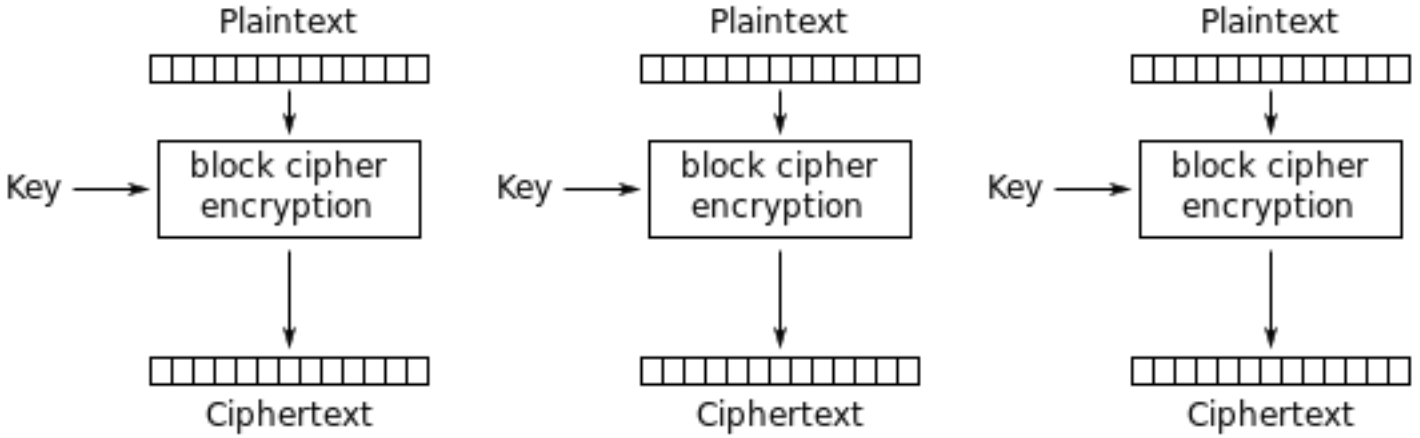
Symmetric Block Cipher - AES

- Advanced Encryption Standard (or Rijndael)
- Superseded DES as the NIST symmetric encryption standard since 2001
- Block size: 128 bits (16 bytes)
- Key size: 128, 192 or 256 bits (slower with longer keys)
- DES vs. AES
  - Longer key size -> more secure
  - AES is faster than DES
  - AES is suitable for parallel processing
- Encryption based on substitution and permutation
  - https://youtu.be/evjFwDRTmV0

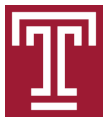


Serious Issue with Block Ciphers

- DES can only process 64-bit blocks
- AES can only process 128-bit blocks
- If you simply divide a long message into 64-bit or 128-bit blocks, and process them independently (this strategy is called "Electronic Codebook")
  - Identical plaintext blocks lead the same ciphertext blocks

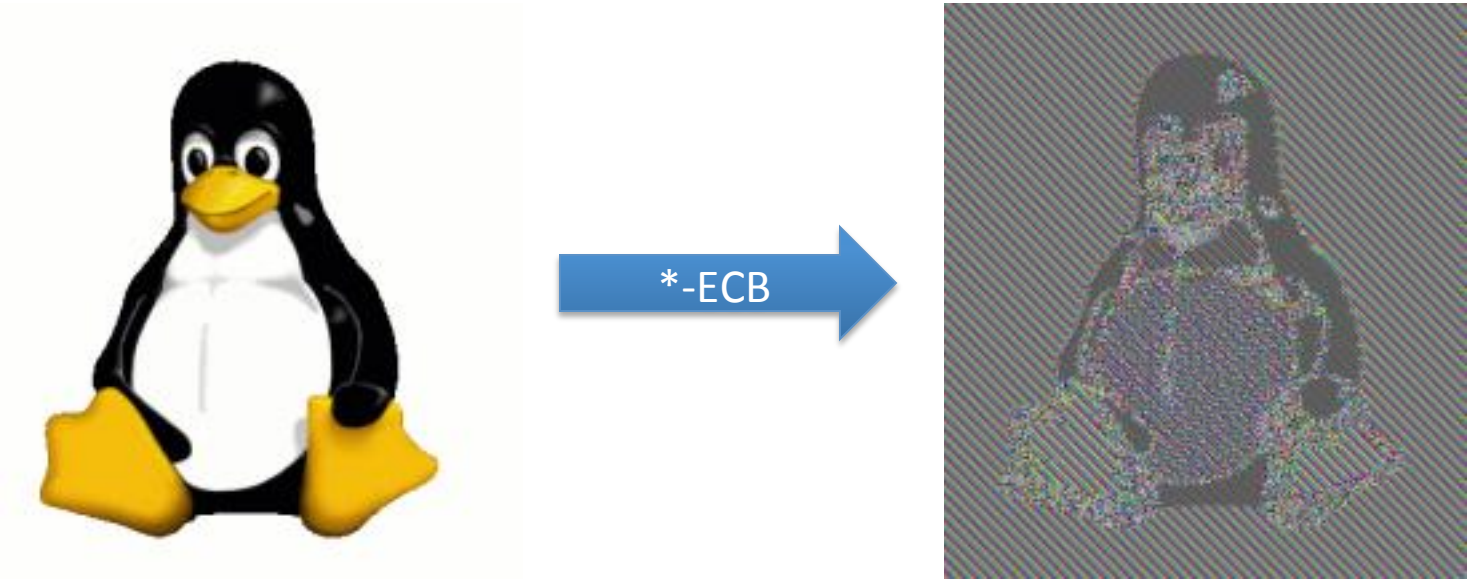


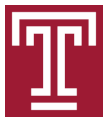
Electronic Codebook (ECB) mode encryption



Serious Issue with Block Ciphers

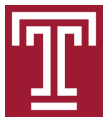
- The simple strategy Electronic Codebook (ECB) leaks too much information
- By applying ECB encryption to the left bitmap image, you get the right one



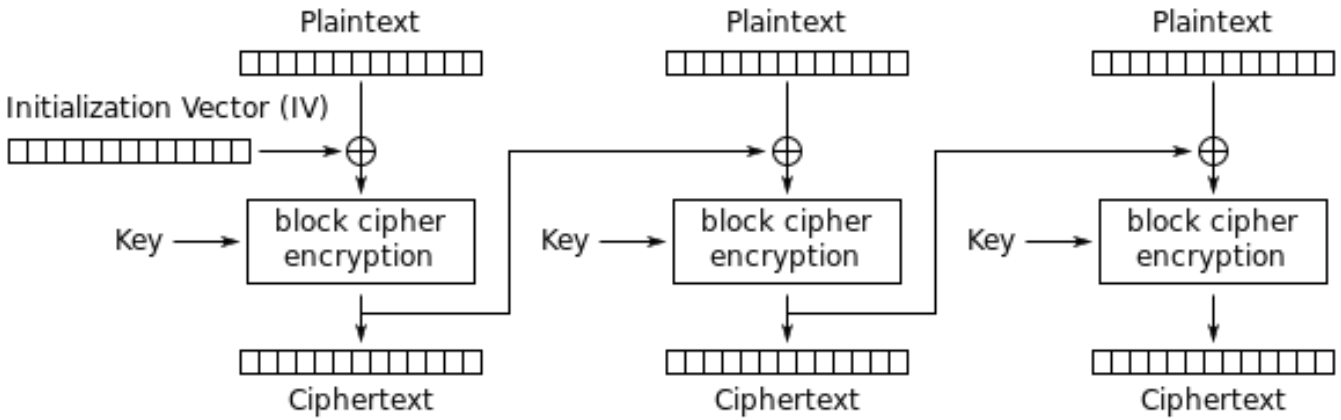


Modes of Operation for Block Ciphers

- A Mode of Operation describes how block ciphers are applied to a message longer than a block; usually, it is simply called Mode
  - E.g., ECB (do not use it), CBC (Cipher Block Chaining), CFB (Cipher Feedback)
- How to interpret "AES128-CBC"?
  - AES128: cipher block with 128-bit key
  - CBC: mode of operation
- Similarly, you can interpret "3DES-CFB"

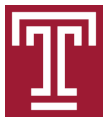


CBC (Cipher Block Chaining)

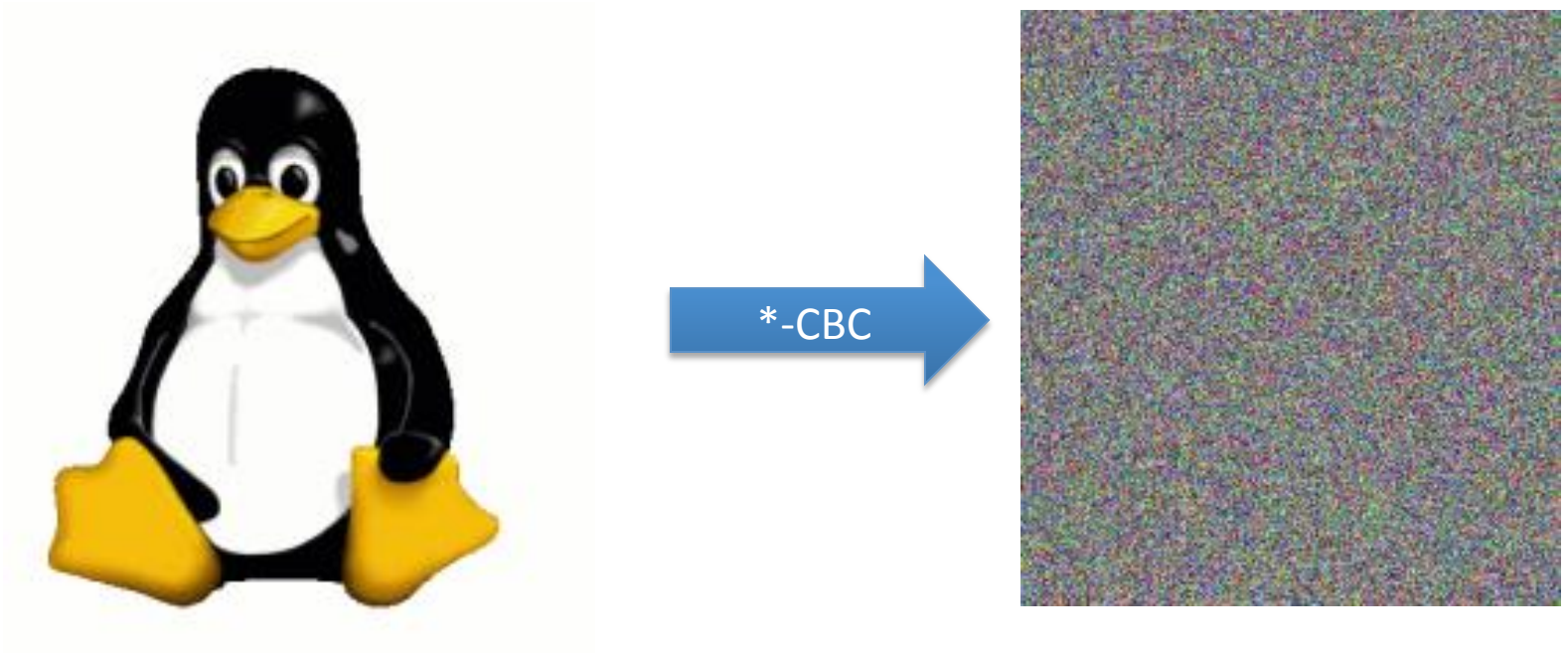


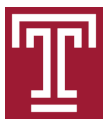
Cipher Block Chaining (CBC) mode encryption

- Even identical plaintext blocks will produce different ciphertext (as long as their preceding ciphertext blocks are different)
- Initialization Vector (IV) has the same size as the plaintext; by varying the IV, even if the same key is applied to encrypting two identical messages, the ciphertext is different
  - Thus the adversary cannot infer whether the two messages are identical

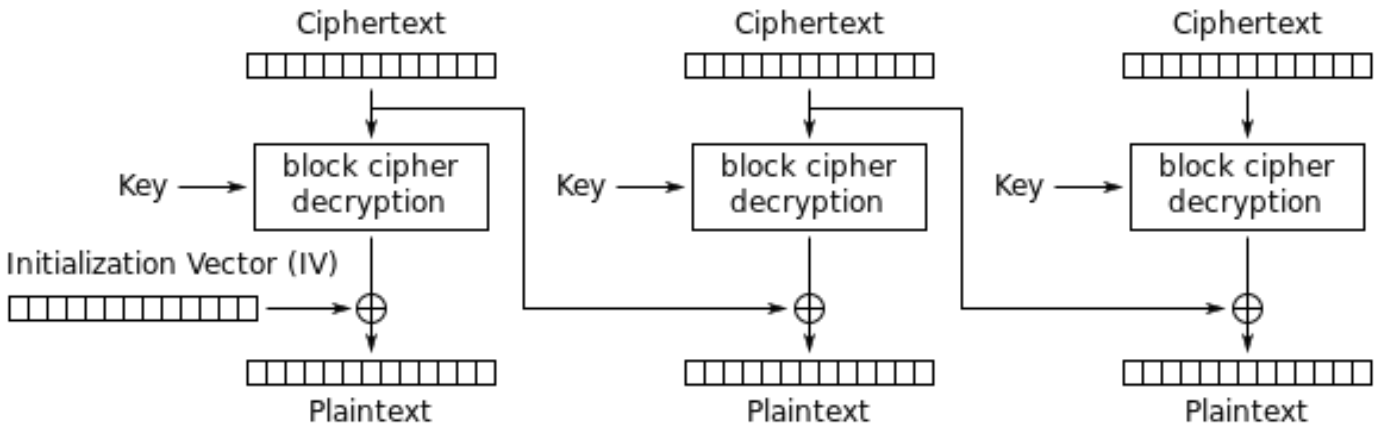


Effect of Applying CBC



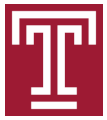


Decryption based on CBC



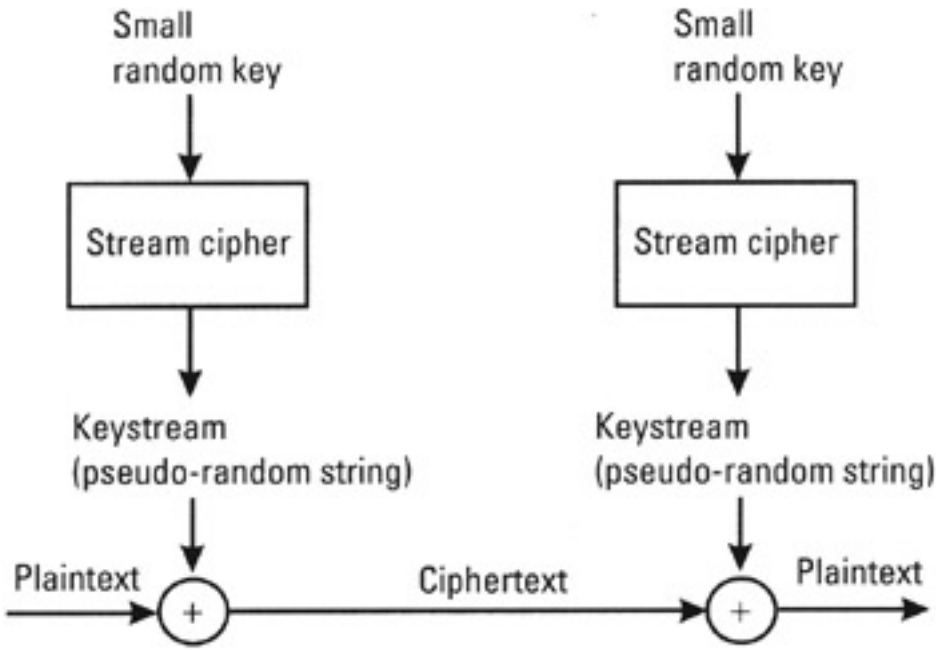
Cipher Block Chaining (CBC) mode decryption

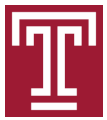
- If you provide an incorrect IV, will you get wrong decryption results?
  - Only the first block is corrupted; you still get correct results for other blocks
  - This property is NOT necessarily true with other modes of operations



Stream Cipher - RC4

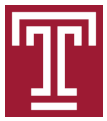
- In stream ciphers, the key is used to generate a keystream (a key-seeded pseudorandom stream of bits), which is XORed with the plaintext





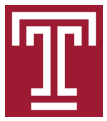
Stream Cipher - RC4

- Rivest Cipher 4 (considered insecure)
- Key size: 40 – 2048 bits
- Warning: The first few bytes of the keystream leak the information of the key; so discard the first, say 1024, bytes of the keystream before using it



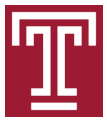
Block Cipher vs. Stream Cipher

- Stream Cipher is faster
- Stream Cipher is simpler to implement in h/w
- Stream Cipher can operate on a single bit
  - In streaming, it is beneficial for reducing the latency
- Stream Cipher works well even if the length of the plaintext is unknown
- Stream Cipher does not need mode of operation
- Stream Cipher does not need padding (e.g., 120-bit data block is padded to 128 bits)



Block Cipher vs. Stream Cipher

- Stream Ciphers have security problems that Block Ciphers do not have
  - Key cannot be reused
    - $C_1 = P_1$ XOR K; $C_2 = P_2$ XOR K;
    - C_1 XOR $C_2 = (P_1$ XOR K) XOR $(P_2$ XOR K) = P_1 XOR P_2
  - Bit-flipping attack
    - Assume P_1 = 1000, P_2 = 9999, $C_1 = P_1$ XOR K
    - You can get $C_2 = P_2$ XOR K by
    - C_1 XOR P_1 XOR $P_2 = (P_1$ XOR K) XOR P_1 XOR $P_2 = P_2$ XOR K

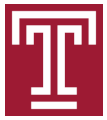


Previous class…

By sending m and Hash(m) on Internet, can the receiver verify the integrity of m, i.e., to verify whether m has been manipulated during the transmission? If not, what should you do?

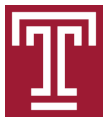
No, cryptographic hash can be used to verify data integrity only if the integrity of the hash value itself is assured. Typically, if the message and the digest value are transmitted in the same channel, it is in vain as the adversary can replace both the message and the corresponding hash value.

Message Authentication Code and Digital Signature are used when the hash value can also be attacked (discussed later)



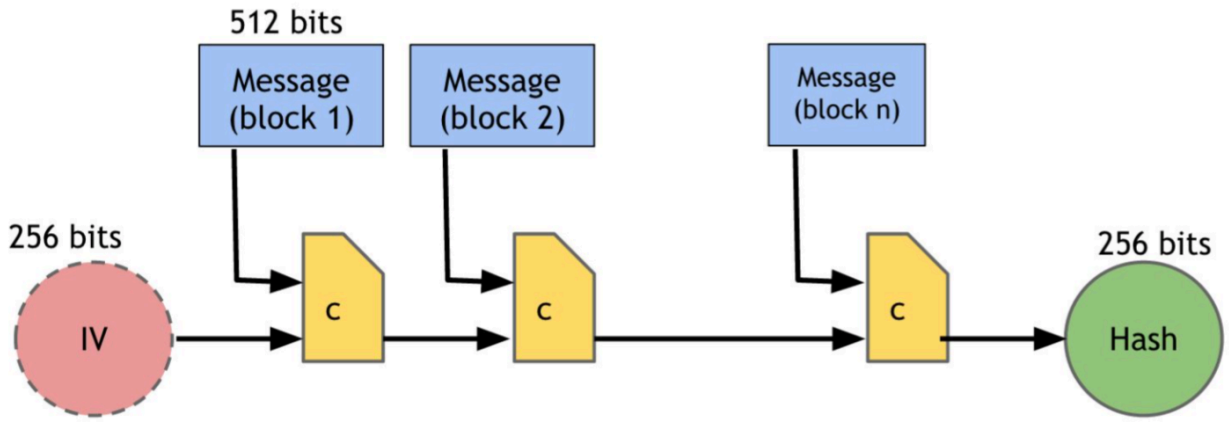
MAC (Message Authentication Code)

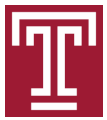
- The digest h = H(m) is generated without any key, such that anyone (including the adversary) can create it
- What if a key is used? Now the adversary cannot forge a keyed-hash value without the key
  - This is the purpose of MAC
- A MAC is a short string used to verify the message integrity and authentication
- mac = MAC(k, m)
  - To prevent the replay attack, the message should contain timestamp, sequence number



A Popular MAC Algorithm: HMAC

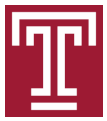
- Hash-based Message Authentication Code
- A typical wrong design:MAC(k, m) = H(k ll m)
  - Susceptible to "*length extension attacks*" if the hash uses the Merkle–Damgård Construction
  - Based on H(k ll m) the adversary can extend the message and still get a valid MAC: H(k ll m ll m')





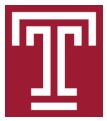
A Popular MAC Algorithm: HMAC

- To resist Length Extension Attack
  - HMAC(m, k) = H(k ll H(k ll m)) // nested hash
- The hash function uses MD5, SHA-1, SHA-2
  - MD5 and SHA-1 are insecure
  - Only SHA-2 is recommended now
  - E.g., if SHA256 is used, it is called HMAC-SHA256
- The key size >= the block size to make full use of the hash resistance
  - In HMAC-SHA256, the key size should be 256 bit

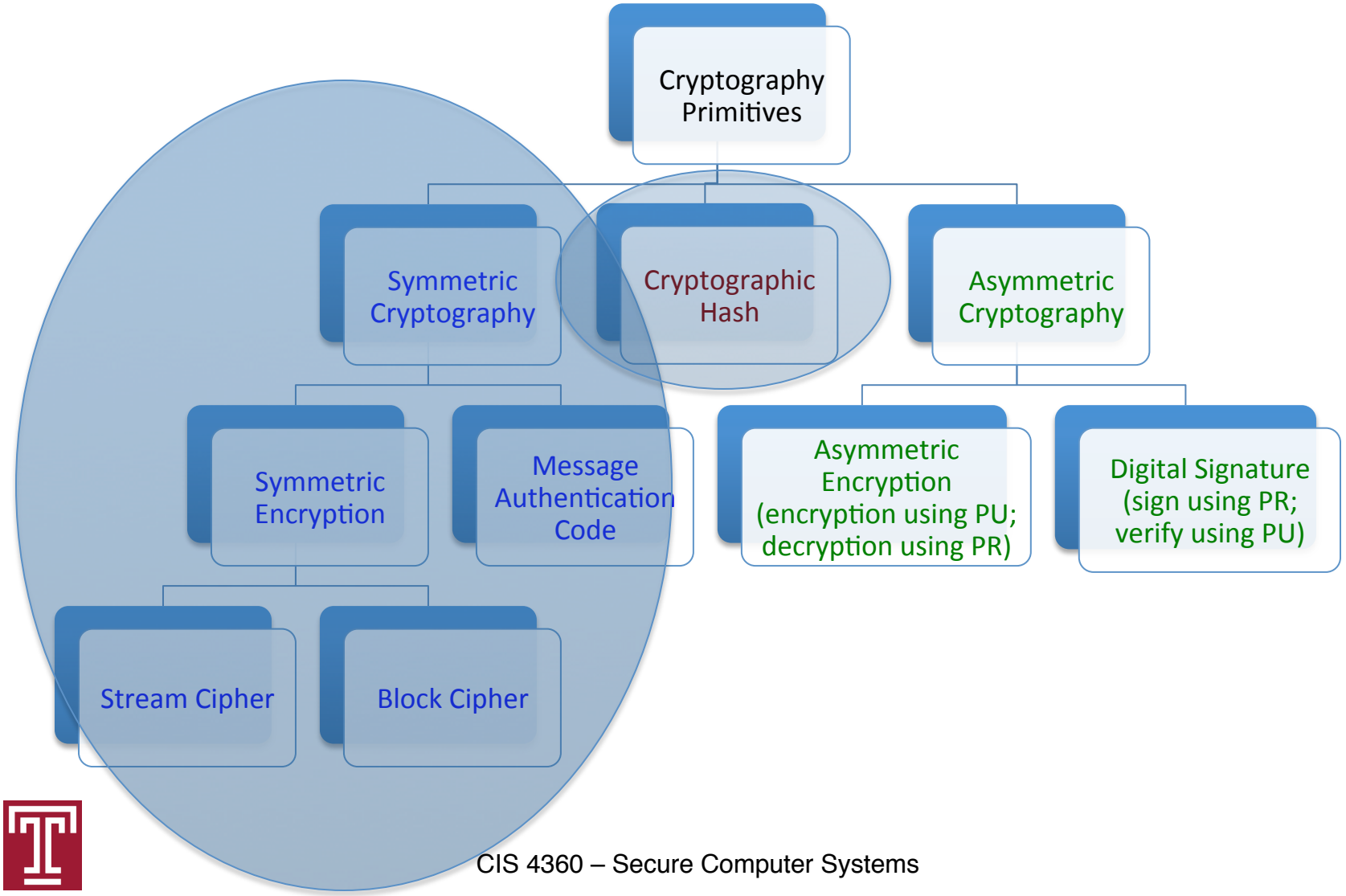


KMAC – SHA-3 Derived MAC

- SHA-3 does *not* use the Merkle-Damgård Construction, so is not vulnerable to the Length Extension Attack
  - Can still use SHA-3 in HMAC (the nested hash), but it is over-kill
- KMAC (Keccak MAC)
  - H(k || m)
  - Uses SHA-3 (Keccak)
  - Simpler and faster than HMAC;
  - Arbitrary-length output
  - NIST in 2016
    - http://csrc.nist.gov/publications/drafts/800-185/sp800_185_draft.pdf



Summary


Cryptography Primitives
Symmetric Cryptography
Cryptographic Hash
Asymmetric Cryptography
Symmetric Encryption
Message Authentication Code
Asymmetric Encryption (encryption using PU; decryption using PR)
Digital Signature (sign using PR; verify using PU)
Stream Cipher
Block Cipher

Writing Assignments

- When to use Stream Ciphers?
- Can MAC be used to achieve non-repudiation?