CS 4910: Into to Computer Security

Cryptographic Tools II

Instructor: Xi Tan

Updates

- Assignment 1 due 2/12
- Project 1 due 2/24

Symmetric Key Cryptography

- So far we've covered:
  - what secure symmetric encryption is
  - high-level design of stream ciphers
  - high-level design of block ciphers
  - DES
  - AES
- Next, we'll talk about:
  - **block** cipher encryption modes and limitations
  - Public Key Cryptography

Block Cipher Modes

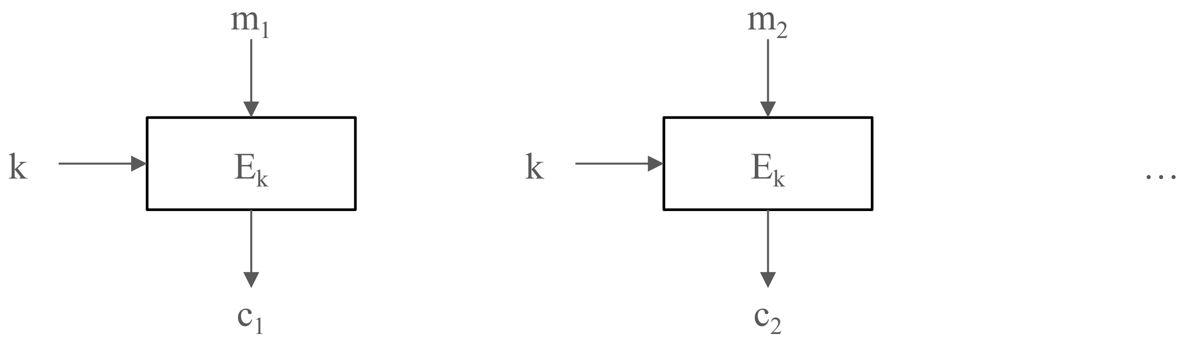
- Encryption modes indicate how messages longer than one block are encrypted and decrypted

- 4 modes of operation were standardized in 1980 for Digital Encryption Standard (DES)
    - electronic codebook mode (ECB), cipher feedback mode (CFB), cipher block chaining mode (CBC), and output feedback mode (OFB)

- 5 modes were specified with the current standard Advanced Encryption Standard (AES) in 2001
    - the 4 above and counter mode (CTR)

Electronic Codebook (ECB) mode

- Electronic Codebook (ECB) mode
  - divide the message m into blocks $m_1 m_2 \ldots m_\ell$ of size n each
  - encrypt each block separately: for $i = 1, \ldots, \ell$, $c_i = E_k(m_i)$, where E denotes block cipher encryption
  - the resulting ciphertext is $c = c_1 c_2 \ldots c_\ell$



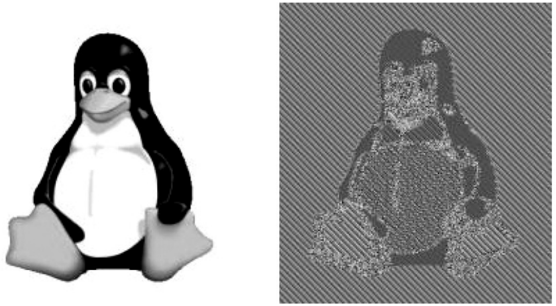
Electronic Codebook (ECB) mode

- Properties of ECB mode:
  - identical plaintext blocks result in identical ciphertexts (under the same key)
  - each block can be encrypted and decrypted independently
  - this mode doesn’t result in secure encryption
- ECB mode is a plain invocation of the block cipher
  - it allows the block cipher to be used in other, more complex
  - cryptographic constructions

Strengths and Weaknesses of ECB

- Strengths:
  - Is very simple
  - Allows for parallel encryptions of the blocks of a plaintext
  - Can tolerate the loss or damage of a block

- Weakness:
  - Documents and images are not suitable for ECB encryption since patterns in the plaintext are repeated in the ciphertext:

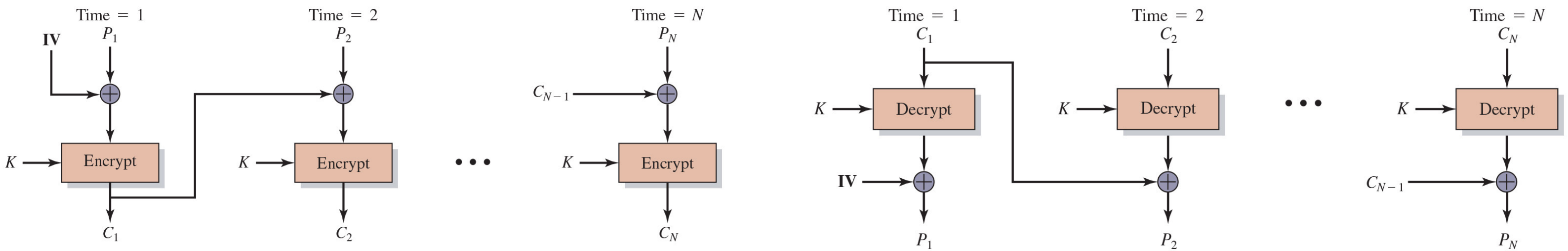


(a) (b)

Figure 8.6: How ECB mode can leave identifiable patterns in a sequence of blocks: (a) An image of Tux the penguin, the Linux mascot. (b) An encryption of the Tux image using ECB mode. (The image in (a) is by Larry Ewing, lewing@isc.tamu.edu, using The Gimp; the image in (b) is by Dr. Juzam. Both are used with permission via attribution.)

Cipher Block Chaining (CBC) Mode

- In Cipher Block Chaining (CBC) Mode
  - The previous ciphertext block is combined with the current plaintext block C[i] = E_K (C[i –1] ⊕ P[i])
  - C[–1] = V, a random block separately transmitted encrypted - known as the Initialization Vector (IV)
  - Decryption: P[i] = C[i –1] ⊕ D_K (C[i])



(a) Encryption

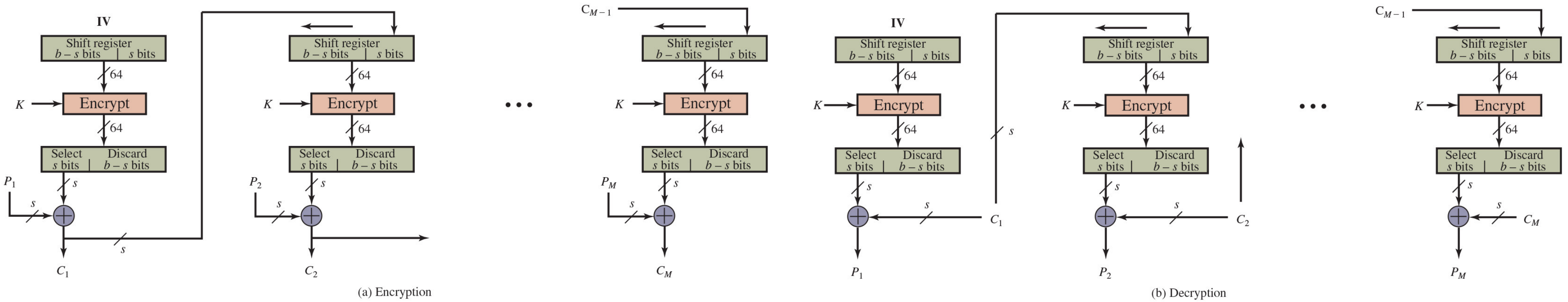
(b) Decryption

Strengths and Weaknesses of CBC

- Strengths:
  - Doesn’t show patterns in the plaintext
  - Is the most common mode
- Weaknesses:
  - CBC requires the reliable transmission of all the blocks sequentially
  - CBC is not suitable for applications that allow packet losses (e.g., music and video streaming)

Cipher Feedback (CFB) mode

- The cipher is given as feedback to the next block of encryption with some new specifications
  - First, an initial vector IV is used for encryption and output bits are divided as a set of s and b-s bits: random IV and set initial input I_1 = IV
  - The left-hand s bits are selected along with plaintext bits to which an XOR operation is applied
  - The result is given as input to a shift register having b-s bits to lhs, s bits to rhs and the process continues
- encryption: $c_i = E_k(I_i) \oplus m_i$; $I_i+1 = c_i$
- decryption: $m_i = c_i \oplus Ek(I_i)$

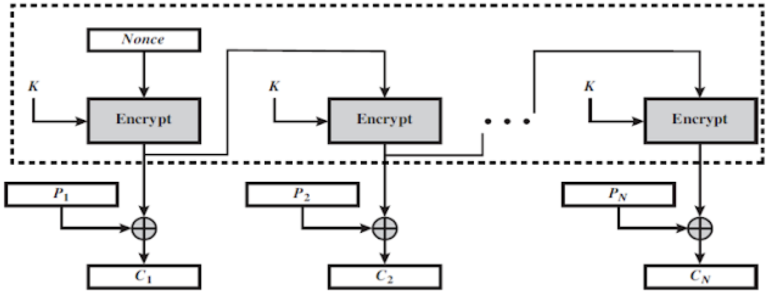


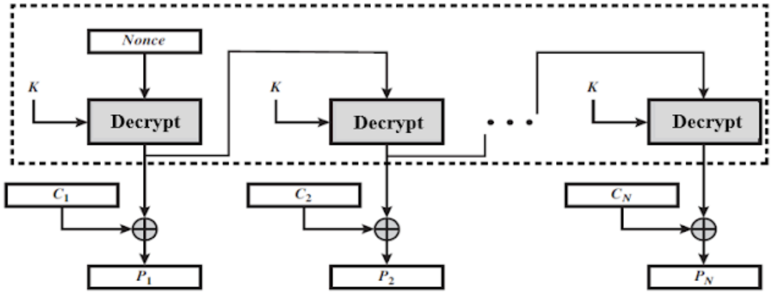
Cipher Feedback (CFB) mode

- This mode allows the block cipher to be used as a stream cipher
  - if our application requires that plaintext units shorter than the block are transmitted without delay, we can use this mode
  - the message is transmitted in r-bit units (r is often 8 or 1)
- Properties of CFB mode:
  - the mode is CPA-secure (under the same assumption that the block cipher is strong)
  - similar to CBC, a ciphertext block depends on all previous plaintext blocks
  - throughput is decreased when the mode is used on small units
  - one encryption operation is applied per r bits, not per n bits

Output Feedback (OFB) mode

- Similar to CFB, but the feedback is from encryption output and is independent of the message
- n-bit feedback is recommended
- using fewer bits for the feedback reduces the size of the cycle





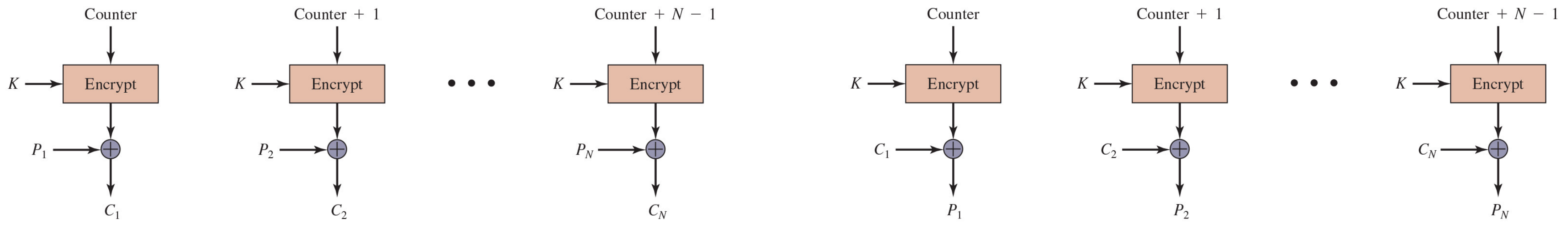
https://www.chiragbhalodia.com/2021/10/ofb-mode.html

Output Feedback (OFB) mode

- Output Feedback (OFB) mode:
  - n-bit feedback is recommended
  - using fewer bits for the feedback reduces the size of the cycle
- Properties of OFB:
  - the mode is CPA-secure
  - the key stream is plaintext-independent
  - similar to CFB, throughput is decreased for $r < n$, but the key stream can be precomputed

Counter (CRT) mode

- a counter is encrypted and XORed with a plaintext block
- no feedback into the encryption function



(a) Encryption (b) Decryption

Counter (CRT) mode

- Advantages of counter mode
  - Hardware and software efficiency: multiple blocks can be encrypted or decrypted in parallel
  - Preprocessing: encryption can be done in advance; the rest is only XOR
  - Random access: its block of plaintext or ciphertext can be processed independently of others
  - Security: at least as secure as other modes
  - Simplicity: doesn’t require decryption or decryption key scheduling
- But what happens if the counter is reused?

Block Cipher Modes of Operation

Mode	Description	Typical Application
Electronic Code book (ECB)	Each block of plaintext bits is encoded independently using the same key.	• Secure transmission of single values (e.g., an encryption key)
Cipher Block Chaining (CBC)	The input to the encryption algorithm is the XOR of the next block of plaintext and the preceding block of ciphertext.	• General-purpose block-oriented transmission • Authentication (CBC-MAC) using last block
Cipher Feedback (CFB)	Input is processed s bits at a time. Preceding ciphertext is used as input to the encryption algorithm to produce pseudorandom output, which is XORed with plaintext to produce next unit of ciphertext.	• General-purpose stream-oriented transmission • Authentication
Output Feedback (OFB)	Similar to CFB, except that the input to the encryption algorithm is the preceding pseudorandom output.	• Stream-oriented transmission over noisy channel (e.g., satellite communication)
Counter (CTR)	Each block of plaintext is XORed with an encrypted counter. The counter is incremented for each subsequent block.	• General-purpose block-oriented transmission • Useful for high-speed requirements

Block Cipher Modes of Operation

Mode	Description	Typical Application
Cipher-based Message Authentication Code (CMAC)	A variation of CBC-MAC mode that adjusts the final message block for greater security before using it to create the final block which is the authentication code.	• Authentication
Offset Codebook (OCB)	Each block of plaintext is XORed with a unique offset before encryption, and an encrypted authentication code is generated.	• Authenticated Encryption on a Stream
Counter with Cipher Block Chaining Mode (CCM)	Provide both confidentiality and the authentication by combining Counter (CTR) and Cipher Block Chaining (CBC-MAC) modes.	• Authenticated Encryption where the data is available in advance
Galois Counter Mode (GCM)	Provide both confidentiality and the authentication by combining Counter (CTR) mode with a Galois polynomial MAC.	• Authenticated Encryption on a Stream

Symmetric Key Cryptography

- **Advantages**
  - Secure, hard to break
    - Offers good confidentiality
  - Fast
    - Many rounds of substitution and transposition
    - Even faster with direct hardware support

- **Disadvantage**
  - The secret key is to be transmitted to the receiving system before the actual message is to be transmitted

Summary for Symmetric Key Cryptography

- AES is the current block cipher standard
  - it offers strong security and fast performance
- Five encryption modes are specified as part of the standard
  - ECB mode is not for secure encryption
  - any other encryption mode achieves sufficient security
    - use one of these modes for encryption even if the message is a single block
- Strong randomness is required for cryptographic purposes
  - key generation, IV generation, etc.

Summary for Symmetric Key Cryptography

- Symmetric encryption principles
  - Cryptography
  - Cryptanalysis
- Data encryption standard (DES)
  - Data encryption standard
  - Triple DES
- Advanced encryption standard (AES)
  - Overview of the algorithm
  - Algorithm details
- Stream ciphers and RC4
  - Stream cipher structure
  - RC4 stream cipher
  - ChaCha20 stream cipher
- Cipher block modes of operation
  - Electronic codebook mode
  - Cipher block chaining mode
  - Cipher feedback mode
  - Counter mode
- Key distribution

What we already know

- Symmetric cryptography tools
  - Stream cipher
  - Block cipher
  - DES, AES
  - Block cipher modes

Next

Cryptographic tools

Public-Key Cryptography

- Public-key encryption
  - a party creates a public-private key pair
    - the public key is pk
    - the private or secret key is sk
- the public key is used for encryption and is publicly available
- the private key is used for decryption only $\mathrm{Dec}_{sk}(\mathrm{Enc}_{pk}(m)) = m$
- knowing the public key and the encryption algorithm only, it is computationally infeasible to find the secret key
- public-key crypto systems are also called asymmetric

Symmetric vs Public Key Cryptography

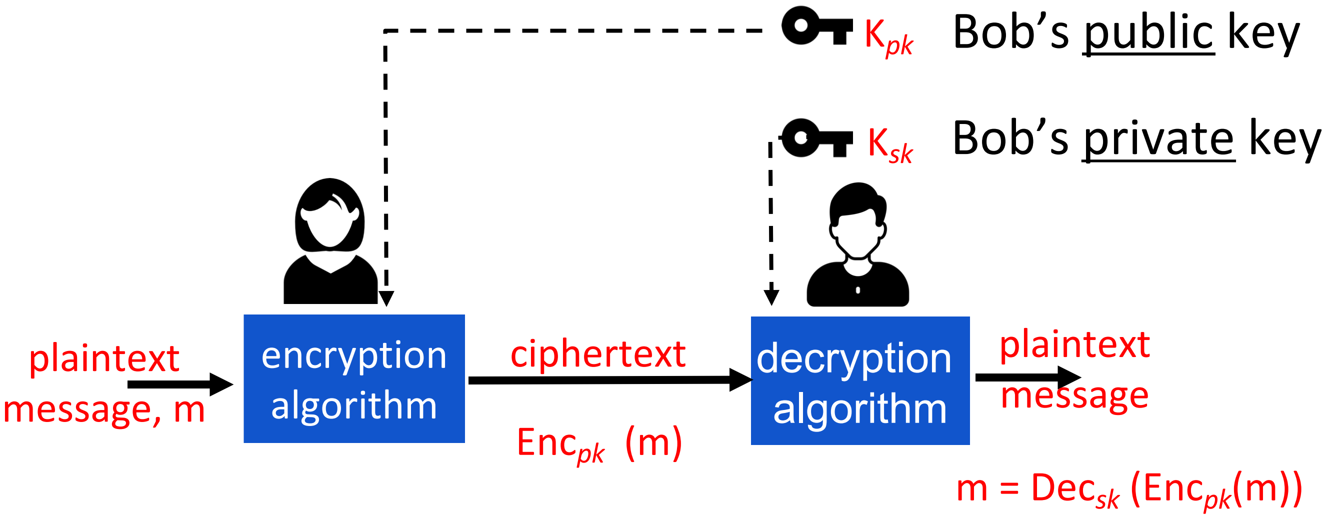
Symmetric key crypto

- requires sender, receiver know shared secret key
- Q: how to agree on key in first place (particularly if never "met")?
  - Challenging for Distributed systems

Public key cryptography

- radically different approach [Diffie-Hellman76, RSA78]
- sender, receiver do *not* share secret key
- *public* encryption key known to *all*
- *private* decryption key known only to receiver

Public Key Cryptography



Public Key Encryption Algorithms

Requirements:

1. need $Enc_{pk}(\cdot)$ and $Dec_{sk}(\cdot)$ such that
   - $Dec_{sk}(Enc_{pk}(m)) = m$
2. given public key K_{pk} , it should be impossible to compute private key K_{sk}

RSA: Rivest, Shamir, Adelson algorithm

Applications for Public-Key Cryptosystems

Algorithm	Digital Signature	Symmetric Key Distribution	Encryption of Secret Keys
RSA	Yes	Yes	Yes
Diffie–Hellman	No	Yes	No
DSS	Yes	No	No
Elliptic Curve	Yes	Yes	Yes

Asymmetric Encryption Algorithms (1 of 2)

- RSA (Rivest, Shamir, Adleman)
  - Developed in 1977
  - Most widely accepted and implemented approach to public-key encryption
  - Block cipher in which the plaintext and ciphertext are
  - integers between 0 and n-1 for some *n*.

- Diffie-Hellman key exchange algorithm
  - Enables two users to securely reach agreement about a shared secret that can be used as a secret key for subsequent symmetric encryption of messages
  - Limited to the exchange of the keys

Asymmetric Encryption Algorithms (2 of 2)

- Digital Signature Standard (DSS)
  - Provides only a digital signature function with SHA-1
  - Cannot be used for encryption or key exchange

- Elliptic curve cryptography (ECC)
  - Security like RSA, but with much smaller keys

Public Key Encryption

- Almost all public-key encryption algorithms use number theory and modular arithmetic
  - RSA is based on the hardness of factoring large numbers
  - ElGamal (based on Diffie-hellman key exchange) is based on the hardness of solving discrete logarithm problem
- RSA is the most commonly used public-key encryption algorithm invented by Rivest, Shamir, and Adleman in 1978
  - sustained many years of attacks on it
  - relies on the fact that factoring large numbers is hard
    - let $n = pq$, where p and q are large primes
    - given only n, it is hard to find p or q, which are used as a trapdoor

RSA: another important property

- The following property will be *very* useful:

$$\mathrm{Dec}_{sk}(\mathrm{Enc}_{pk}(m)) = m = \mathrm{Enc}_{pk}(\mathrm{Dec}_{sk}(m))$$

use public key first, followed by private key

use private key first, followed by public key

Result is the same!

RSA Public-Key Encryption

- Uses exponentiation of integers modulo a prime
- RSA key generation
  - generate two large prime numbers p and q of the same length
  - Compute n = pq
  - Choose a small prime number e
  - Compute the smallest d such that ed mod (p-1)(q-1) = 1
- public key PU = {e, n}
- private key PR = {d, n}

RSA Public-Key Encryption

- Both sender and receiver know values of *n* and *e*
- Only receiver knows value of *d*
- Encryption:
  - given a message m such that $0 < m < n$
  - given a public key pk = (e, n)
  - Encrypt as $C = Enc_{pk}(m) = m^e \bmod n$
- Decryption:
  - given a ciphertext c ($0 < c < n$)
  - given a public key pk = (e, n) and the corresponding private key sk = d
  - Decrypt as $m = Dec_{sk}(c) = c^d \bmod n = (m^e)^d \bmod n = m$

Plain RSA Encryption

- Example of Plain RSA
  - key generation
    - p = 11, q = 7, n = pq = 77, ϕ(n) = 60
    - e = 37 ⇒ d = 13 (i.e., ed = 481; ed mod 60 = 1)
    - public key is pk = (37, 77) and private key is sk = 13
  - Encryption
    - let m = 15
    - c = Enc(m) = m^e mod n = 1537 mod 77 = 71
  - Decryption
    - m = Dec(c) = c^d mod n = 7113 mod 77 = 15

Security of RSA

Existing attacks on RSA:

Brute force

Involves trying all possible private keys

Mathematical attacks

There are several approaches, all equivalent in effort to factoring the product of two primes

Timing attacks

These depend on the running time of the decryption algorithm

Chosen ciphertext attacks

This type of attack exploits properties of the R S A algorithm

Timing Attack Countermeasures

- Constant exponentiation time
  - Ensure that all exponentiations take the same amount of time before returning a result
  - This is a simple fix but does degrade performance
- Random delay
  - Better performance could be achieved by adding a random delay to the exponentiation algorithm to confuse the timing attack
  - If defenders do not add enough noise, attackers could still succeed by collecting additional measurements to compensate for the random delays
- Blinding
  - Multiply the ciphertext by a random number before performing exponentiation
  - This process prevents the attacker from knowing what ciphertext bits are being processed inside the computer and, therefore, prevents the bit-by-bit analysis essential to the timing attack

Towards Safe Use of RSA

- Padded RSA
  - plain RSA is deterministic
  - this is even worse than in case of symmetric encryption
    - anyone can search for m encrypting various messages
  - we can randomize ciphertext by padding each m with random bits
    - now a message can be at most k – t bits long
    - random t bits are added to it such that 2^t work is infeasible

Towards Safe Use of RSA

- PKCS #1 v1.5 was a widely used standard for padded RSA
  - PKCS = RSA Laboratories Public-Key Cryptography Standard
  - it is believed to be CPA-secure
- PKCS #1 v2.0 utilizes OAEP (Optimal Asymmetric Encryption Padding)
  - the newer version mitigates some attacks on v1.5 and is known to be CCA-secure
  - in OAEP, we use plain RSA encryption on $m \oplus g(r) \| r \oplus h(m \oplus g(r))$, where h and g are hash functions and r is randomness

Towards Safe Use of RSA

- Making factoring infeasible
  - choose n to be long enough (we can choose any n!)
  - for a security parameter k, compute n with $|n| = k$

- A good implementation will also have countermeasures against implementation-level attacks
  - timing attacks, special cases of e and d, etc.

Symmetric vs Public-Key Encryption

- Public-key operations are orders of magnitude slower than symmetric encryption
  - a multiplication modulo n requires close to O(|n|2) work
  - a full-size exponentiation modulo n requires close to O(|n|3) work
    - it is the cost of multiplication times the exponent size
  - public-key encryption is typically not used to communicate large volumes of data
    - it is rather used to communicate (or agree on) a symmetric key
    - the data itself is sent encrypted with the symmetric key
- In RSA, decryption is significantly slower than encryption, with key generation being the slowest

Next

Cryptographic tools

- Overview
- Symmetric Key Cryptography
- Public Key Cryptography
    - Diffie-Hellman Key Exchange
- Message Integrity and Digital Signatures
- Summary