Data Security and Privacy

Overview of Symmetric Cryptography
Books That Might Be Of Interest

THE CODE BOOK
THE SCIENCE OF SECRECY FROM ANCIENT EGYPT TO QUANTUM CRYPTOGRAPHY
NATIONAL BESTSELLER
"It would be hard to imagine a clearer or more fascinating presentation... Mr. Singh gives cryptography not only its historical dimension but its human face."
--THE NEW YORK TIMES
SIMON SINGH
Bestselling Author of FERMAT'S ENIGMA

APPLIED CRYPTOGRAPHY
20TH ANNIVERSARY EDITION
Protocols, Algorithms, and Source Code in C
BRUCE SCHNEIER
WILEY
### Goals of Cryptography

- The most fundamental problem cryptography addresses: **ensure security of communication over insecure medium**

- **What does secure communication mean?**
  - confidentiality (secrecy)
    - only the intended recipient can see the communication
  - integrity (authenticity)
    - the communication is generated by the alleged sender

- **What does insecure medium mean?**
  - Two basic possibilities:
    - Passive attacker: the adversary can eavesdrop
    - Active attacker: the adversary has full control over the communication channel
Basic Terminology for Encryption

- **Plaintext**
  - original message

- **Ciphertext**
  - transformed message

- **Key**
  - secret used in transformation

- **Encryption**

- **Decryption**

- **Cipher**
  - algorithm for encryption/decryption
Mono-alphabetic Substitution Cipher

- The key space: all permutations of $\Sigma = \{A, B, C, \ldots, Z\}$
- Encryption given a key $\pi$:
  - each letter $X$ in the plaintext $P$ is replaced with $\pi(X)$
- Decryption given a key $\pi$:
  - each letter $Y$ in the cipherext $P$ is replaced with $\pi^{-1}(Y)$

Example:

\begin{align*}
A & B C D E F G H I J K L M N O P Q R S T U V W X Y Z \\
\pi = & B A D C Z H W Y G O Q X S V T R N M L K J I P F E U
\end{align*}

**BECAUSE** $\rightarrow$ **AZDBJSZ**
Cryptanalysis of Substitution Ciphers: Frequency Analysis

- **Basic ideas:**
  - Each language has certain features: frequency of letters, or of groups of two or more letters.
  - Substitution ciphers preserve the language features.
  - Substitution ciphers are vulnerable to frequency analysis attacks.
How to Defeat Frequency Analysis?

• Use different substitutions to get rid of frequency features.
  – Leads to polyalphabetic ciphers, one-time pad, and stream ciphers such as RC4

• Use larger blocks as the basis of substitution. Rather than substituting one letter at a time, substitute 64 bits at a time, or 128 bits.
  – Leads to block ciphers such as DES & AES.
One-Time Pad

- Key is a random string that is at least as long as the plaintext
- Encryption is similar to shift cipher
- Invented by Vernam in the 1920s
One-Time Pad

Let $Z_m = \{0, 1, \ldots, m-1\}$ be the alphabet.

Plaintext space = Ciphertext space = Key space = $(Z_m)^n$

The key is chosen uniformly randomly

Plaintext $X = (x_1 \ x_2 \ \ldots \ x_n)$

Key $K = (k_1 \ k_2 \ \ldots \ k_n)$

Ciphertext $Y = (y_1 \ y_2 \ \ldots \ y_n)$

$e_k(X) = (x_1+k_1 \ x_2+k_2 \ \ldots \ x_n+k_n) \mod m$

$d_k(Y) = (y_1-k_1 \ y_2-k_2 \ \ldots \ y_n-k_n) \mod m$
Shannon (Information-Theoretic) Security = Perfect Secrecy

Basic Idea: Ciphertext should reveal no “information” about Plaintext

Definition. An encryption over a message space $\mathcal{M}$ is perfectly secure if

- $\forall$ probability distribution over $\mathcal{M}$
- $\forall$ message $m \in \mathcal{M}$
- $\forall$ ciphertext $c \in \mathcal{C}$ for which $\Pr[C=c] > 0$

We have

$$\Pr[PT=m \mid CT=c] = \Pr[PT = m].$$
Explanation of the Definition

- \( \Pr [PT = m] \) is what the adversary believes the probability that the plaintext is \( m \), before seeing the ciphertext.
- \( \Pr [PT = m \mid CT = c] \) is what the adversary believes after seeing that the ciphertext is \( c \).
- \( \Pr [PT = m \mid CT = c] = \Pr [PT = m] \) means that after knowing that the ciphertext is \( C_0 \), the adversary’s belief does not change.
An Equivalent Definition of Perfect Secrecy

Definition. An encryption scheme is perfectly secure if and only if for any ciphertext c, and any two plaintext m1 and m2, the probability that m1 is encrypted to c is the same as the probability that m2 is encrypted to c.

\[ \forall \text{ message } m1, m2 \]
\[ \forall \text{ ciphertext } c \]
\[ \Pr [CT = c \mid PT = m1] = \Pr [CT = c \mid PT = m2] \]
Perfect Secrecy

- Fact: When keys are uniformly chosen in a cipher, the cipher has perfect secrecy iff. the number of keys encrypting M to C is the same for any (M,C)
  - This implies that
    \[ \forall c \forall m_1 \forall m_2 \Pr[\text{CT}=c \mid \text{PT}=m_1] = \Pr[\text{CT}=c \mid \text{PT}=m_2] \]

- One-time pad has perfect secrecy when limited to messages over the same length (Proof?)
Key Randomness in One-Time Pad

• One-Time Pad uses a very long key, what if the key is not chosen randomly, instead, texts from, e.g., a book are used as keys.
  – this is not One-Time Pad anymore
  – this does not have perfect secrecy
  – this can be broken
  – How?

• The key in One-Time Pad should never be reused.
  – If it is reused, it is Two-Time Pad, and is insecure!
  – Why?
Usage of One-Time Pad

- To use one-time pad, one must have keys as long as the messages.
- To send messages totaling certain size, sender and receiver must agree on a shared secret key of that size.
  - typically by sending the key over a secure channel
- Key agreement is difficult to do in practice.
- Can’t one use the channel for sending the key to send the messages instead?
- Why is OTP still useful, even though difficult to use?
Usage of One-Time Pad

• The channel for distributing keys may exist at a different time from when one has messages to send.

• The channel for distributing keys may have the property that keys can be leaked, but such leakage will be detected
  – Such as in Quantum cryptography
The “Bad News” Theorem for Perfect Secrecy

- Question: OTP requires key as long as messages, is this an inherent requirement for achieving perfect secrecy?
- Answer. Yes. Perfect secrecy implies that key-length $\geq$ msg-length

Proof:

- Implication: Perfect secrecy difficult to achieve in practice
Stream Ciphers

- In One-Time Pad, a key is a random string of length at least the same as the message.

- Stream ciphers:
  - Idea: replace “rand” by “pseudo rand”
  - Use Pseudo Random Number Generator
  - PRNG: \(\{0,1\}^s \rightarrow \{0,1\}^n\)
    - expand a short (e.g., 128-bit) random seed into a long (e.g., \(10^6\) bit) string that “looks random”
  - Secret key is the seed
  - Basic encryption method: \(E_{key}[M] = M \oplus \text{PRNG}(\text{key})\)
Pseudo Random Number Generator

- Useful for cryptography, simulation, randomized algorithm, etc.
  - Stream ciphers, generating session keys
- The same seed always gives the same output stream
  - Why is this necessary for stream ciphers?
- Simulation requires uniform distributed sequences
  - E.g., having a number of statistical properties
- Cryptographically secure pseudo-random number generator requires unpredictable sequences
  - satisfies the "next-bit test": given consecutive sequence of bits output (but not seed), next bit must be hard to predict
- Some PRNG’s are weak: knowing output sequence of sufficient length, can recover key.
  - Do not use these for cryptographic purposes
Security Properties of Stream Ciphers

- Under known plaintext, chosen plaintext, or chosen ciphertext, the adversary knows the key stream (i.e., PRNG(key))
  - Security depends on PRNG
  - PRNG must be “unpredictable”

- Do stream ciphers have perfect secrecy?

- How to break a stream cipher in a brute-force way?

- If the same key stream is used twice, then easy to break.
  - This is a fundamental weakness of stream ciphers; it exists even if the PRNG used in the ciphers is strong
Using Stream Ciphers in Practice

• If the same key stream is used twice, then easy to break.
  – This is a fundamental weakness of stream ciphers; it exists even if the PRNG used in the ciphers is strong

• In practice, one key is used to encrypt many messages
  – Example: Wireless communication
  – Solution: Use Initial vectors (IV).
  – $E_{key}[M] = [IV, M \oplus PRNG(key || IV)]$
    • IV is sent in clear to receiver;
    • IV needs integrity protection, but not confidentiality protection
    • IV ensures that key streams do not repeat, but does not increase cost of brute-force attacks
    • Without key, knowing IV still cannot decrypt
  – Need to ensure that IV never repeats! How?
Towards Computational Security

• Perfect secrecy is too difficult to achieve.
• Computational security uses two relaxations:
  – Security is preserved only against \textit{efficient}
    (computationally bounded) adversaries
    • Adversary can only run in feasible amount of time
  – Adversaries can potentially succeed with some \textit{very small probability} (that we can ignore the case it actually happens)
Defining Security

• Desire “semantic security”, i.e., having access to the ciphertext does not help adversary to compute any function of the plaintext.
  – Difficult to use

• Equivalent notion: Adversary cannot distinguish between the ciphertexts of two plaintexts
Towards IND-CPA Security:

- Ciphertext Indistinguishability under a Chosen-Plaintext Attack: Define the following IND-CPA experiment:
  - Involving an Adversary and a Challenger
  - Instantiated with an Adversary algorithm $A$, and an encryption scheme $\Pi = (\text{Gen}, \text{Enc}, \text{Dec})$

Challenger

$k \leftarrow \text{Gen}()$

$\choose m_0, m_1 \in M$

$b \leftarrow_R \{0, 1\}$

$C = \text{Enc}_k[m_b]$

$\choose b' \in \{0, 1\}$

Adversary

chooses $m_0, m_1 \in M$

$\text{Adversary wins if } b = b'$
Computational Security vs. Information Theoretic Security

• If a cipher has only computational security, then it can be broken by a brute force attack, e.g., enumerating all possible keys
  – Weak algorithms can be broken with much less time

• How to prove computational security?
  – Assume that some problems are hard (requires a lot of computational resources to solve), then show that breaking security means solving the problem

• Computational security is foundation of modern cryptography.
Why Block Ciphers?

• One thread of defeating frequency analysis
  – Use different keys in different locations
  – Example: one-time pad, stream ciphers

• Another way to defeat frequency analysis
  – Make the unit of transformation larger, rather than encrypting letter by letter, encrypting block by block
  – Example: block cipher
Block Ciphers

- An n-bit plaintext is encrypted to an n-bit ciphertext
  - \( P : \{0,1\}^n \)
  - \( C : \{0,1\}^n \)
  - \( K : \{0,1\}^s \)
  - \( E : K \times P \rightarrow C : E_k : \text{a permutation on } \{0,1\}^n \)
  - \( D : K \times C \rightarrow P : D_k \text{ is } E_k^{-1} \)
- Block size: \( n \)
- Key size: \( s \)
Data Encryption Standard (DES)

- Designed by IBM, with modifications proposed by the National Security Agency
- US national standard from 1977 to 2001
- De facto standard
- Block size is 64 bits;
- Key size is 56 bits
- Has 16 rounds
- Designed mostly for hardware implementations
  - Software implementation is somewhat slow
- Considered insecure now
  - Vulnerable to brute-force attacks
- Triple DES: $E_{k_3}D_{k_2}E_{k_1}(M)$ has 112-bit strength, but slow
Advanced Encryption Standard

• In 1997, NIST made a formal call for algorithms stipulating that the AES would specify an unclassified, publicly disclosed encryption algorithm, available royalty-free, worldwide.
• Goal: replace DES for both government and private-sector encryption.
• The algorithm must implement symmetric key cryptography as a block cipher and (at a minimum) support block sizes of 128-bits and key sizes of 128-, 192-, and 256-bits.
• In 1998, NIST selected 15 AES candidate algorithms.
• On October 2, 2000, NIST selected Rijndael (invented by Joan Daemen and Vincent Rijmen) to as the AES.
Need for Encryption Modes

- A block cipher encrypts only one block
- Needs a way to extend it to encrypt an arbitrarily long message
- Want to ensure that if the block cipher is secure, then the encryption is secure
- Aims at providing Semantic Security (IND-CPA) assuming that the underlying block ciphers are strong
Block Cipher Encryption Modes: ECB

- Message is broken into independent blocks;
- **Electronic Code Book (ECB)**: each block encrypted separately.

- Encryption: $c_i = E_k(x_i)$
- Decryption: $x_i = D_k(c_i)$
Properties of ECB

- **Deterministic:**
  - the same data block gets encrypted the same way,
    - reveals patterns of data when a data block repeats
  - when the same key is used, the same message is encrypted the same way

- **Usage:** not recommended to encrypt more than one block of data

- **How to break the semantic security (IND-CPA) of a block cipher with ECB?**
DES Encryption Modes: CBC

- **Cipher Block Chaining (CBC):**
  - Uses a random Initial Vector (IV)
  - Next input depends upon previous output

  **Encryption:** $C_i = E_k (M_i \oplus C_{i-1})$, with $C_0 = IV$

  **Decryption:** $M_i = C_{i-1} \oplus D_k(C_i)$, with $C_0 = IV$
Properties of CBC

• Randomized encryption: repeated text gets mapped to different encrypted data.
  – can be proven to provide IND-CPA assuming that the block cipher is secure (i.e., it is a Pseudo Random Permutation (PRP)) and that IV’s are randomly chosen and the IV space is large enough (at least 64 bits)

• Each ciphertext block depends on all preceding plaintext blocks.

• Usage: chooses random IV and protects the integrity of IV
  – The IV is not secret (it is part of ciphertext)
  – The adversary cannot control the IV
Encryption Modes: CTR

- **Counter Mode (CTR):** Defines a stream cipher using a block cipher
  - Uses a random IV, known as the counter
  - Encryption: $C_0=\text{IV}$, $C_i=M_i \oplus E_k[\text{IV}+i]$
  - Decryption: $\text{IV}=C_0$, $M_i=C_i \oplus E_k[\text{IV}+i]$
Properties of CTR

• Gives a stream cipher from a block cipher

• Randomized encryption:
  – when starting counter is chosen randomly

• Random Access: encryption and decryption of a block can be done in random order, very useful for hard-disk encryption.
  – E.g., when one block changes, re-encryption only needs to encrypt that block. In CBC, all later blocks also need to change
Data Integrity and Source Authentication

- Encryption does not protect data from modification by another party.
- Most encryption schemes are **malleable**:  
  - Modifying ciphertext result in (somewhat) predictable change in plaintext
- Need a way to ensure that data arrives at destination in its original form as sent by the sender.
Hash Functions

• A hash function maps a message of an arbitrary length to a m-bit output
  – output known as the fingerprint or the message digest

• What is an example of hash functions?
  – Give a hash function that maps Strings to integers in $[0, 2^{32}-1]$

• Cryptographic hash functions are hash functions with additional security requirements
Security Requirements for Cryptographic Hash Functions

Given a function \( h : X \rightarrow Y \), then we say that \( h \) is:

- **preimage resistant (one-way):**
  if given \( y \in Y \) it is computationally infeasible to find a value \( x \in X \) s.t. \( h(x) = y \)

- **2-nd preimage resistant (weak collision resistant):**
  if given \( x \in X \) it is computationally infeasible to find a value \( x' \in X, x' \neq x \) and \( h(x') = h(x) \)

- **collision resistant (strong collision resistant):**
  if it is computationally infeasible to find two distinct values \( x', x \in X \), s.t. \( h(x') = h(x) \)
Usages of Cryptographic Hash Functions

• Software integrity
  – E.g., tripwire

• Timestamping (cryptographic commitment)
  – How to prove that you have discovered a secret on an earlier date without disclosing the context of a secret?

• Other applications
  – Message authentication
  – One-time passwords
  – Digital signature
Choosing Parameters

- The level of security (for collision resistance) of a hash function that outputs $n$ bits, is about $n/2$ bits
  - i.e., it takes $2^{n/2}$ time to bruteforce it
  - Assuming that no better way of attacking the hash function is known
- Longer outputs often means more computation time and more communication overhead
- The level of security for encryption function using $k$-bit key is about $k$ bits
Choosing the length of Hash outputs

• **The Weakest Link Principle:**
  – A system is only as secure as its weakest link.
  – Hence all links in a system should have similar levels of security.

• Because of the birthday attack, the length of hash outputs in general should double the key length of block ciphers
  – SHA-224 matches the 112-bit strength of triple-DES (encryption 3 times using DES)
  – SHA-256, SHA-384, SHA-512 match the new key lengths (128,192,256) in AES
  – If small output size is highly important, and one is sure that collision-resistance is not needed (only one-wayness is needed), then same size should be okay.
Well Known Hash Functions

- **MD5**
  - output 128 bits
  - collision resistance completely broken by researchers in China in 2004

- **SHA1**
  - output 160 bits
  - In Feb 2017, collision found with $\approx 10^{19} \approx 2^{63}$ SHA-1 evaluations (costing around $\$100K$ using cloud)
  - one-wayness still holds

- **SHA2 (SHA-224, SHA-256, SHA-384, SHA-512)**
  - outputs 224, 256, 384, and 512 bits, respectively
  - No practical security concerns yet
Using Hash Functions for Message Integrity

- Method 1: Uses a Hash Function $h$, assuming an authentic (adversary cannot modify) channel for short messages
  - Transmit a message $M$ over the normal (insecure) channel
  - Transmit the message digest $h(M)$ over the authentic channel
  - When receiver receives both $M'$ and $h$, how does the receiver check to make sure the message has not been modified?

- This is insecure. How to attack it?
- A hash function is a many-to-one function, so collisions can happen.
Hash Family

• A hash family is a four-tuple \((X, Y, K, H)\), where
  – \(X\) is a set of possible messages
  – \(Y\) is a finite set of possible message digests
  – \(K\) is the keyspace
  – For each \(K \in K\), there is a hash function \(h_K \in H\). Each \(h_K : X \rightarrow Y\)

• Alternatively, one can think of \(H\) as a function \(K \times X \rightarrow Y\)
Message Authentication Code (MAC)

- A MAC scheme is a hash family, used for message authentication
- \( \text{MAC}(K, M) = H_K(M) \)
- The sender and the receiver share secret \( K \)
- The sender sends \((M, H_K(M))\)
- The receiver receives \((X, Y)\) and verifies that \(H_K(X) = Y\), if so, then accepts the message as from the sender
- To be secure, an adversary shouldn’t be able to come up with \((X', Y')\) such that \(H_K(X') = Y'\).

MAC: Using a shared secret (or a limit-bandwidth confidential channel) to achieve authenticity/integrity.
Security Requirements for MAC

• Secure against the “Existential Forgery under Chosen Plaintext Attack”
  – Challenger chooses a random key K
  – Adversary chooses a number of messages $M_1$, $M_2$, ..., $M_n$, and obtains $t_j = \text{MAC}(K,M_j)$ for $1 \leq j \leq n$
  – Adversary outputs $M'$ and $t'$
  – Adversary wins if $\forall j \ M' \neq M_j$, and $t' = \text{MAC}(K,M')$

• Basically, adversary cannot create the MAC value for a message for which it hasn’t seen an MAC
Constructing MAC from Hash Functions

• Let $h$ be a one-way hash function

• $\text{MAC}(K,M) = h(K || M)$, where $||$ denote concatenation
  – Insecure as MAC with a hash function that uses the Merkle-Damgard construction:
  – given $M$ and $t = h(K || M)$, adversary can compute $M' = M || \text{Pad}(M) || X$ and $t'$, such that $h(K || M') = t'$
HMAC: Constructing MAC from Cryptographic Hash Functions

\[ \text{HMAC}_K[M] = \text{Hash}[(K^+ \oplus \text{opad}) || \text{Hash}[(K^+ \oplus \text{ipad})||M])] \]

- K^+ is the key padded (with 0) to B bytes, the input block size of the hash function
- ipad = the byte 0x36 repeated B times
- opad = the byte 0x5C repeated B times.

At high level, \( \text{HMAC}_K[M] = H(K || H(K || M)) \)

Hash function is used twice, in nested fashion.
HMAC Security

- If used with a secure hash functions (e.g., SHA-256) and according to the specification (key size, and use correct output), no known practical attacks against HMAC
Next Lecture

• Overview of Public-Key Cryptography