

CS2107 Introduction to Information Security

AY23/24 Sem 2, github.com/gerteck

0. Introduction

CS2107: Introductory module, illustrates fundamentals of how systems fail due to malicious activities and how they can be protected.

Lecture Notes notation

- **Textbook:** Security in Computing (5th ed). Prentice Hall. Reference [PFx.y] refer to chapter x section y of this book.
- Links with “read”: Part of lecture, required.
- Links with “see”: Optional good to browse references.

Internet Security Threat Report Exc. Summary

Persistent threats are threats that often operate within the shadows, outside of attention focused.

- Highly targeted, often use combination of social engineering and software vulnerabilities to establish footholds within the targeted enterprise.
- Network protection not enough to mitigate threats, comprehensive monitoring program required to scan all internal and external network traffic. Identifying, securing data is also key to protecting assets.
- Recent attack types: Formjacking, Ransomware, Living off the Land (LoL),

Introduction to Computer/Info Security

Systems may fail due to operator mistakes, hardware failures, poor implementation etc. Many systems robust against typical noise. Security is about intentional failures inflicted by deliberate human actions.

Security Definitions: C-I-A triad

- **Confidentiality:** Prevention of unauthorized disclosure of information.
- **Integrity:** Prevention of unauthorized modification of info / processes.
- **Availability:** Prevention of unauthorized withholding of info / resources.
- Other requirements may include (Confidentiality: anonymity, covert channel), (Integrity: Non-repudiation (digital signature)), (Other: Accountability, traitor-tracing (printout w hidden watermark)) etc.
- Importance of understanding security requirements before adopting mechanisms. Do not adopt mismatched protection mechanisms.

Difficulties in achieving Security

- **Security not considered** (in early design stage). Lack of adversarial thinking in design / trade-off in security with ease-of-use, performance / cost.
- Difficult to formulate requirements /design / implementation bugs.
- Difficult to operate/manage (Human error).
- **Known Vulnerabilities: CVE (Common vulnerabilities & Exposures)**. Repository of discovered vulnerabilities. Significant portion considered “implementation bugs”.
- **Zero-day Vulnerabilities:** Unpublished vulnerabilities. If attackers deploy attacks on zero-day v., victims have “zero-day” to react. Not easy to get.

1. Encryption

Key Summary & Takeaways

- **Encryption designed for confidentiality.** (Not necessarily integrity).
- Formulate attack scenario by defining attacks it can prevent.
- Notions of “Oracle”: Encryption, Decryption, Padding Oracle.
- **Key strength:** Quantifying security by equivalence of best-known attack to exhaustive search.
- No known efficient attacks on modern schemes under “original” threat models, but there are pitfalls. Such as implementation error (wrong mode, wrong random source, mishandle of IV), side channel attack, implicitly require integrity, padding oracle attack.
- Design of various symmetric key encryption schemes:
 - One-time pad, stream cipher (xor’ing with “pseudo-random” string)
 - Block cipher (mode of operations: CBC, ECB, CTR, GCM)
- **Crucial role of IV.** (Need randomness for indistinguishability). Make encryption probabilistic, how it is deployed, why it is important.

Definitions

- **Symmetric Key Encryption Scheme:** Two algorithms: encryption and decryption. Meets correctness property, and must be secure.
- **Correctness:** $(D_k(E_k(x)) = x)$. **Security:** Informally, “difficult” to derive useful info of the key k, and plaintext x. Ciphertext resemble random sequence of bytes.
- **Cryptography:** Study of techniques in securing communication in present of adversaries with access. Encryption just a primitive (other includes crypto hash, digital signature etc. Common placeholders: Alice, Bob, Eve (“eavesdropper”), Mallory (malicious, modify message), etc.

Attack Model

Aka: Threat / Adversary / Security Model, Attack Scenario.

Measuring **security of a system**: Through attack classes it can prevent. Secure w.r.t these classes of attacks. Attack models are application-dependent. Attack models described by:

- Attacker’s knowledge (info / service exposed) and computing resources
- Attacker’s goal

Types of information we assume access to: (Access to info can be formulated to accesses to an “Oracle”).

- **Ciphertext only attack:** Adversary given collection of ciphertext c. May know some properties of the plaintext, for e.g. the plaintext is an English sentence. (adv. can’t choose plaintext).
- **Known plaintext attack:** Adversary given collection of plaintext m and corresponding ciphertext c. (adv. can’t choose plaintext)
- **Chosen plaintext attack (CPA):** Access to blackbox (i.e. the oracle). Can choose, feed any plaintext m, obtain corresponding ciphertext c (all encrypt with the same key), reasonable large number of times. Can see ciphertext and choose next input. (*black-box called encryption oracle*).
- **Chosen ciphertext attack (CCA2):** Same as chosen plaintext attack, but adv. chooses ciphertext and blackbox outputs plaintext. (*black-box called decryption oracle*).
- **Note:** (strange to assume attacker has power to decrypt, but good reason is that in some practical scenario, attacker does have some but not full decryption capability, e.g. *padding oracle*). To cater for all scenarios, in formulation and design of encryption, we consider oracle with full decryption capability.)

Adversary’s Goals

- **Total Break:** Attacker wants to find the key.
- **Partial Break:** Few definitions, e.g. decrypt ciphertext, determine coarse information about plaintext, etc.
- **Indistinguishability (IND):**) Attacker may satisfy with distinguishability of ciphertext: with some “non-negligible” probability more than $\frac{1}{2}$, the attacker is able to distinguish the ciphertexts of a given plaintext (say, “Y”) from the ciphertext of another given plaintext (say, “N”). (Equivalently, unable to distinguish the ciphertext and a randomly sequence).
- **Note:** total break most difficult goal, since achieves all. Distinguishability weakest goal, design cryptosystem preventing weakest goal.

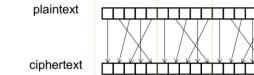
Classical Ciphers (Symmetric)

1. Substitution Cipher

- **Plaintext, ciphertext:** String over a set of symbols U.
- **Key:** Substitution table S, 1-1 onto function from U to U.
- **Key space:** Set of all possible keys (e.g. 27!). **Space Size** is total number of possible keys (factorial, 27!). **Key Size** is number of bits required to represent a key. ($\log_2(27!)$, since 27! unique)
- **Exhaustive Search (Brute-force): Can be infeasible in worst case**
- **Known-plaintext-attack:** Access to pairs of ciphertext and corresponding plaintexts: *Sub. cipher “not secure / totally broken under known plaintext attack”*. (Possible in practice, e.g. certain words in header of email “From, Subject”, protocols bytes fixed header information etc.)
- **Ciphertext only attack:** Vulnerable to frequency analysis attack, when plaintexts English sentences.

2. Permutation Cipher (aka transposition cipher)

- Groups plaintext into blocks of t characters, applies secret permutation (1-1 onto function). Fails miserably on known-plaintext attack.



- S & P cipher not secure, performing substitution multiple times no use. However, by interlacing them (S&P), attacks become more difficult. Many modern encryption scheme (e.g. AES) designed using rounds of S & P.

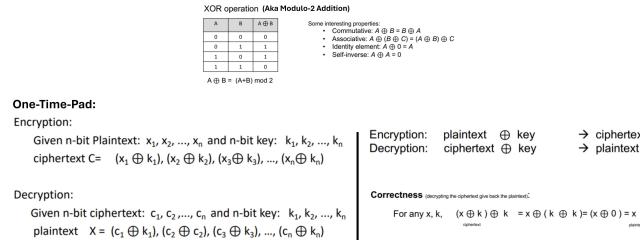
Terminology

- **Cryptosystem:** A system for encryption and decryption.
- **Plaintext:** Original form of message.
- **Ciphertext:** Encrypted form of message.
- **Perfect Secrecy:** A cryptosystem has perfect secrecy if for any distribution X , for all x, y :
$$Pr(X = x|Y = y) = Pr(X = x)$$
- For any ciphertext y and plaintext x , the chances attacker correctly predicts x before knowing y , and after knowing y , are the same.
- **Work Factor:** Difficulty of breaking an encryption (Amount of effort necessary).
 - (E.g. determine time it would take to test single password, multiply by total possible passwords).

3. One Time Pad

One-time pad (OTP) is an encryption technique that requires the use of a single-use pre-shared key that is larger than or equal to the size of the message being sent.

- Plaintext is paired with a random secret key (known as one-time pad).
- Each bit or character of the plaintext encrypted by combining it with corresponding bit/character from pad using modular addition.[1]



- **Requirement:** Key cannot be re-used, used only once. Hence, 1GB plaintext would need 1GB key to encrypt.
- From pair of ciphertext & plaintext, key can be derived. However, key useless, as only used once.
- **Security:** OTP leaks no information of plaintext, sometimes called "unbreakable". There is a exhaustive key for any English sentence. (Perfect Secrecy)

Modern Ciphers (Symmetric)

Generally refers to schemes that use computer to encrypt / decrypt. E.g:

- DES [Data Encryption Standard 1977]
- RC4 [Rivest's Cipher 4 1987]
- A5/1 [used in GSM 1987]: Used in WEP Wifi 1999, switched to WPA2
- AES [Adv. Encrypt. Std.], RSA.

- Designs take into consideration of known-plaintext-attack, freq. analysis and other known attacks.
- Supposed to be secure, so any successful attack does not perform noticeably better than exhaustive search.

Key Length, Exhaustive Search DES, AES

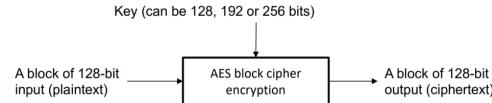
- **Security of Encryption Scheme:** Quantified by **length of key**, w.r.t. exhaustive search.
- Given a key length of 32 bits, there are 2^{32} possible keys. Hence, exhaustive search needs to "loop" 2^{32} in worst case.
- **No. of bits to be considered "secure":** 128, 192, 256 bits,
- (NIST recommendation for AES).

1. DES, AES and Exhaustive Search

- **Exhaustive Search on DES:** Key length of DES is 56 bits. Previously, seemed infeasible, but has since become easily broken.
- **AES:** New standard for block cipher in 2000, AES block length is 127, key length can be 128, 192 or 256 bits.
- Currently, no known attacks on AES. NSA classifies AES as "Suite B Cryptography".

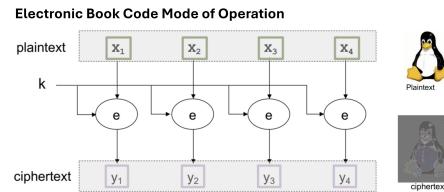
2. Block Cipher & Mode-of-Operations

- **Block Cipher:** DES & AES known as "Block Cipher". Block cipher designed for some fixed size input/output.
 - E.g. AES designed for 128 bits input/output.
- **Block Cipher Mode Of Operation:** Describes how to repeatedly apply cipher's single block operation to securely transform amounts of data larger than a block. (Extending encryption from single block to multiple blocks).



Mode-of-Operation: ECB Mode on AES

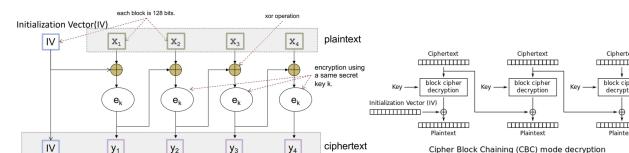
- **Electronic Book Code:** Divide plaintext into blocks, apply block cipher to each block, with the same key.
- ECB leaks information! AES encryption deterministic without randomly chosen IV.
- **Deterministic:** Produces same ciphertext for given plaintext and key over separate executions.



Mode-of-Operation: CBC Mode on AES

- **Cipher Block Chaining on AES:** Using an IV, uses chaining process that causes decryption of block of ciphertext to depend on all preceding ciphertext blocks.
- **Initialization Vector (IV):** Arbitrary number of certain length used with secret key, to provide the initial state, for data encryption.
- **Encryption:** Each plaintext block is XOR-ed with previous ciphertext block, and then encrypted. Process repeats until all plaintext is ciphertext blocks.
- **Decryption:** Reverse encryption process. Note, process does not need to start with final block, can happen simultaneously as all inputs present.

Cipher Block Chaining Mode of Operation

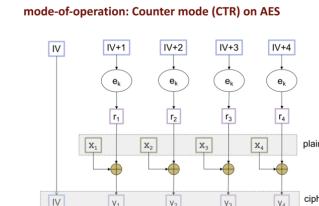


- The Initialization Vector (IV) is an arbitrary value chosen during encryption. So, is different in different encryptions of the same plaintext. Depending on implementation, IV can be randomly chosen, obtain from a counter, or from other info.

Note: In the above figure, we treat IV as part of the final ciphertext. The terminology is not consistent in the literature. Some documents may state that "the final message to be sent are the IV and the ciphertext" (i.e. IV is not included in the "ciphertext"). In this module, when it is crucial, we will explicitly state whether IV is included or excluded. (e.g. "Ciphertext together with an IV").

Mode-of-Operation: Counter Mode (CTR) on AES

- **Counter Mode:** Turns block cipher into stream cipher, generates next keystream block by encrypting successive values of a "counter". Counter can be any function producing sequence, incld. simple increment by one counter.



Mode-of-Operation: GCM Mode (Galois/Counter)

- **GCM:** Combines Counter mode (CTR) with Galois authentication. Construction of mode more complicated, is "Authenticated-Encryption".
- Ciphertext consists of extra tag for authentication, secure in presence of decryption oracle.

Python Programming: CBC

• Python.

(package PyCryptodome <https://pycryptodome.readthedocs.io/en/latest/src/cipher/aes.html>)

```
>>> from Crypto.Cipher import AES
>>> key = b'Sixteen-byte key'
>>> iv = b'Sixteen-byte IV'
>>> cipher = AES.new(key, AES.MODE_CBC, iv)
>>> c = cipher.encrypt(b'Plaintext of length with multiple of 16 bytes')
```

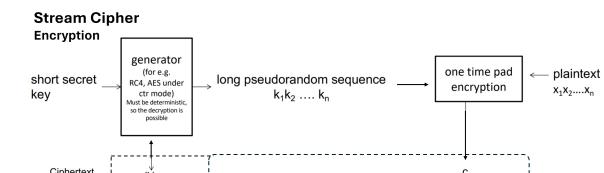
In Python, to display a byte sequence, we can use...

```
>>> from base64 import *
>>> b16encode(c)
b'536978465658206279746520204956b186083256C4CCBD1638AF4877FB2AAFBBCB66FE13C403D7CE8E8A04D028E66CA6E1294
F51C2F9363CCB95313TA6A3'
```

3. Stream Cipher and IVs

Stream Cipher

- **Stream Cipher:** A symmetric key cipher where plaintext combined with **pseudorandom cipher digit stream (keystream)**.
- "Inspired" by one-time-pad, generate some cryptographically secure pseudorandom sequence.
- w/o knowing secret key, computationally diff. to distinguish from truly random sequence. Similarly diff. to get short secret key from sequence, or predict part of sequence from another part.
- IV: Most ciphers have **Initialization Vector**, randomly chosen or from counter.



Decryption: Given the key and the ciphertext with the IV.
Step 1: Extracts the IV from the ciphertext.
Step 2: From the key and IV, generates the long sequence.
Step 3: Performs xor to get the plaintext.

Role of Unique IV

- Recall, IV appended to front of c to form ciphertext. This IV must be different for every message. IV need not be secret.
- Sequence is derived from IV and secret key. IV needs to be unique for every message! Otherwise, leaks information.
- Unique IV:** If IV different, two pseudorandom sequences will be different. Two ciphertexts of same plaintext will be different. Can just randomly choose the IV for each message.
- Hence, xor'ing two ciphertexts will not cancel out pseudosequences.
- IV makes encryption “probabilistic.”**

Why IV? What if the IV is always the same?

Suppose there isn't an IV (or the IV is always set to be a string of 0's)

Consider the situation where the same key is used to encrypt two different plaintexts

$$X = x_1, x_2, x_3, x_4, x_5 \text{ and}$$

$$Y = y_1, y_2, y_3, y_4, y_5$$

Further suppose that an attacker eavesdropped and obtained the two corresponding ciphertexts, U, V.

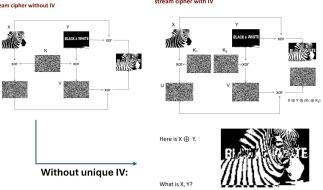
The attacker can now compute

$$U \oplus Y = (X \oplus K) \oplus (Y \oplus K)$$

By associative and commutative property of xor

$$U \oplus Y = (X \oplus Y) \oplus (K \oplus K) = X \oplus Y.$$

So, from U and V, the attackers can obtain information about $X \oplus Y$, i.e. the following sequence

$$(U \oplus Y), (x_1 \oplus y_1), (x_2 \oplus y_2), (x_3 \oplus y_3), (x_4 \oplus y_4), (x_5 \oplus y_5)$$


- IV also needed in **CBC mode**, to make encryption non-deterministic. If encryption deterministic (without IV), it will leak information on whether plaintext of two ciphertext are the same, if $C_1 == C_2$. Could be crucial piece of information.

Types of Attacks

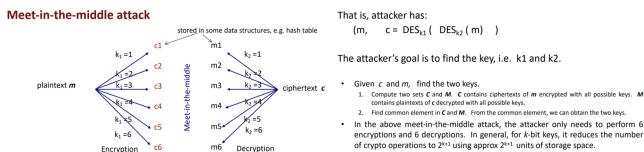
Meet-In-The-Middle Attack (Double / Triple DES)

Double / Triple DES

- Double / Triple DES:** To improve DES security, (DES weak as key length of 56 bits short), do multiple repeated encryption using different keys. A reason for this may be to utilize already existing suitable hardware to encrypt.
- DES does not form a group ($E_{k1}(E_{k2}(x)) \neq E_{k2}(x)$) for some $k3$, so makes sense to use multiple encryption.
- Double DES:** Consider double encryption, we expect key-strength to be 112 . (2^{56+56}). However, attacker, with storage space, can reduce key strength to below 57.

Meet-In-The-Middle Attack

- MITM:** Meet-in-the-middle Attack, a well-known plaintext attack, is a generic space-time tradeoff cryptographic attack against encryption schemes that perform multiple encryption operations in sequence.
- Breaking two-part encryption from both sides simultaneously.
- Primary reason why **Double DES not used**, and why **Triple DES key** (168-bit) can be brute forced by attacker with 2^{56} space and 2^{122} operations.
- Mechanism:** Assume attack has a pair (m, c) of plaintext and corresponding ciphertext.
- Remedy:** Use triple encryption, but with 2 keys. (E.g. 3DES, TDES, 3TDES etc.)



Padding Oracle Attack

Padding Format

- For fixed size blocks, e.g. block size of AES is 128bits (16 bytes), padding is needed to fit plaintext into last block.
- Many ways to fill value, but important piece of information encoded: **number of padded bits**. If information missing, receiver will not know length of plaintext.
- E.g. **PKCS#7 Padding Standard**.

PKCS#7 Padding Standard

- PKCS#7 is a padding standard.
- Read [https://en.wikipedia.org/wiki/Padding_\(cryptography\)#PKCS7](https://en.wikipedia.org/wiki/Padding_(cryptography)#PKCS7)

- The following example is self-explanatory.
Suppose the block size is 8 bytes, and the last block has 5 bytes (thus 3 extra bytes required), padding is done as follow:

DD DD DD DD DD DD DD DD 03 03 03

- In general, the paddings are:

01
02 02
03 03 03
04 04 04 04 etc.

- If the last block is full, i.e. it has 8 bytes, an extra block of all zeros is added.

Oracle

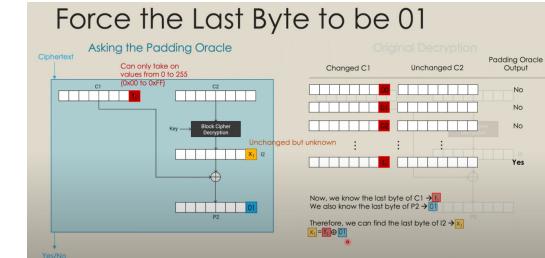
- Security Analysis:** Need to know 1. **information attackers have**, 2. attacker's goals.
- Oracle:** Query-answer system. Attacker can send in multiple queries, Oracle will output the answer. E.g.
 - Encryption Oracle:** Output ciphertext for given plaintext, of s. key k .
 - Decryption Oracle:** Output plaintext given ciphertext, of s. key k .
- Padding Oracle:** Attacker can query a ciphertext (encrypted using some secret key k , padding oracle knows k). Oracle outputs yes/no if plaintext is in correct “padding” format.
- Padding Oracle can come in many forms, e.g. query response behavior, response time, subtle differences.
- Padding Oracle Attack Model:**
 - Attacker has ciphertext including iv: (iv, c)
 - Attack's goal: plaintext of (iv, c) .

Padding Oracle Attack (on AES CBC Mode)

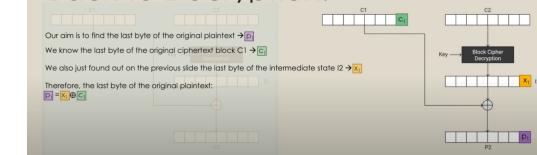
- AES CBC mode not secure against padding oracle attack. (In particular, when done with PKCS#7).
- Easily extend algorithm to find all plaintext. Attack is practical as there are protocols between client and server which performs this. *Now, if attack obtained ciphertext, attack can interact with server to get plaintext.*
- Prevention of Padding Oracle Attack:**
 - Deny access to such Oracle. Might not be feasible all the time.
 - Change padding standard to mitigate attack. However, may be smarter way to attack new padding.
 - Using CTR mode might avoid padding. In practice, bit strings have to be padded in one way or another.
 - Padding Oracle weaker form of Decryption oracle. If scheme secure in presence of decryption oracle, scheme also secure against padding oracle attack. GCM believed to be IND-CCA2 secure.

Padding Oracle Attack Algorithm

- Algorithm:**
 - Force last X bytes to be padding
 - Do this by modifying byte of previous ciphertext/IV (Loop till YES)
 - Here, by XOR'ing padding and input, we get intermediate byte.
 - By XOR'ing this int. byte with original IV / ciphertext, find plaintext!
- Easily extend algorithm** to find all plaintext, by using increasing length of padding, decrypt from end to start. (Right to Left, repeat process).



Back to Decryption!



Cryptography Pitfalls

- Any secure encryption scheme can be vulnerable if not implemented or adopted properly. Examples include:
 - Predictable Randomness:** Secret key generated predictable, compromise security
 - Modify/Design own encryption scheme:** “Don’t roll your own crypto /security protocol”. Use standard library.

Wrong choice of IV

- IV Generation:** IV's must not be same for two different ciphertexts.
- E.g. To encrypt file F, IV derived from filename/meta-data.
- E.g. **Microsoft RC4 [implementation] Flaw**, in both Word and Excel, where same IV used when document modified, causing part of documents recovered with negligible amount of computation.
- When using AES under CBC mode, IV should be unpredictable to prevent certain attack. (E.g. IV = 1, 2, 3). BEAST attack exploits this.
- Reusing one-time pad:** E.g. US Verona Project against Soviets.

Reliance on Obscurity: Kerchoff's Principle

- Principle:** System should be secure even if everything about system, except secret key, is public knowledge.
- Security through Obscurity:** To hide design of system to achieve security. Not advisable to reveal certain settings (network structure, settings, algorithm, implementation, usernames etc). Obscurity as additional layer in defense-in-depth strategy. Deter, discourage novice attackers.
- E.g. **Against relying on Obscurity:** RC4 algorithm was trade secret, but was anonymously posted. MIFARE Classic smartcard also reversed-engineered.

2. Authentication Credential

- **Authentication Credential** as any data (PIN, digi. certificate, password) or device that is issued to individual / system used to authenticate identity for purposes of facilitating access.
- **Authentication:** Process of assuring communicating entity or origin of piece of information is one it claims to be.
- **Authenticity implies integrity.**



• Authentication Process:

- For **data-origin auth**, one way is to use crypto scheme such as Signature, or MAC (message auth. code).
- For **communication entity auth**, need some authentication protocol employing above crypto primitives.
- **Credential:** Unforgeable info required for authentication. (E.g. password is an authentication credential.)

2.1 Password

Password System

1. **Bootstrapping:** User, server establish common password, server keeps file recording identity (*userid, username*) & *password*.
- (Bootstrapping, establish user chosen /or default password.)
2. **Authentication:** Server authenticates entity. Entity who gives correct password to claimed identity authentic.
3. **Password Reset:** Many reasons to reset password.

Weak Authentication System, Replay Attack

- Simple sending of identity and password protocol is "**weak authentication**" system.
- Such protocol subjected to simple "**replay attack**".
- Information sniffed from communication channel, replay to impersonate.
- Under **strong authentication**, info sniffed cannot be used to impersonate.

Attacks on Password System

- **Attack bootstrapping:** Attacks make use of default passwords.
- **Attack password reset process:** *Security-Cost Usability tradeoff*.
 - **Fallback authentication:** Security Qn / Pw. reset.
 - Enhance usability, reduce cost but reduce security.
 - **Danger:** Social engineering + pw. reset.
- **Search for correct password:**
 - **Exhaustive search:** Test all combinations
 - **Dictionary Attacks:** Online & Offline dictionary attacks.
- **Dictionary attacks** Two scenarios in dictionary attacks:
 - **Online dictionary attack:** an attacker must interact with the authentication system during the searching process. In other words, attacker must be online. (e.g. choose a password and ask the system (oracle) whether it is authentic)
 - **Offline dictionary attack:** There are two phases.
 1. The attacker obtains some information **D** about the password, possibly via some interactions. (e.g. somehow the hash of the password is sent over the protocol.)
 2. Next, the attacker carries out the searches using **D** without interacting with the system. e.g. testing hash of words in dictionary.

• Stealing of password:

1. **Sniffing:** Shoulder surfing (look-over-shoulder attack), Sniffing communication (uncommon now, sniffing unencrypted password over public network in clear).
2. **Viruses, Keylogger:** Computer viruses as key-loggers, or hardware keyloggers. Send captured data back to attack via "covert channel".
3. **Phishing:** Trick to voluntarily send password to attacker. Social engineering attack.
 - **Spear Phishing:** targeted attack against particular small group of users.
 - **Phishing, Pharming, Vishing, Smishing**
 - **Prevention:** User training, blacklisting.
4. **Cache:** Shared workstation where information is cached.
5. Lost of password file

Password Strength

• Password Entropy: Measure of (randomness) password strength.

Remark: Password Entropy

We often encounter this term "entropy" when quantifying strength of password. Entropy is a measurement of randomness. In this class we won't go into the definition of entropy. We can use the following example to have a sense of its meaning.

Suppose a set P contains N unique passwords. Alice chooses her passwords by randomly & uniformly picking a password from the set P . Every password in P has an equal chance to be chosen (i.e. 1/ N). In this case, by definition, the entropy of Alice's password is: $(\log_2 N)$ bits

What if Alice doesn't choose the passwords uniformly, for e.g., the probability that she picks a word starting with letter "A" is higher than the probability that she picks a word starting with "Z"? In such cases, the entropy is not $(\log_2 N)$. By the definition of entropy, it is

$$\sum_{i=1}^N p_i \log_2 p_i$$

where p_i is the probability that Alice picks the i -th word in P . (If we put $p_i = 1/N$, then we get $\log_2 N$.)

It can be shown that, for the entropy to be highest for a set of N items, p_i must be $1/N$. In other words, uniform choices.

(omit if this further confuse you) Another way to measure randomness is min-entropy, which is

$$\min_i(-\log_2 p_i)$$

Suppose with probability 0.5, Alice picks her password as "Alice", and for probability 0.5, she uniformly chooses from P . That is, each word in P has probability $1/(2N)$ being chosen, and "Alice" has 0.5. Now, the entropy is roughly $(\log_2 N - 1)$, which is high. However, there is good chance in correctly guessed her password. So, entropy might not be a good measure of password strength. Note that in this case, the min-entropy is low and is 1, correctly reflects the poor choice.

- Truly random password high "entropy", difficult to remember. User selection biased. Human generated 20-digit password likely to attain entropy much less than $20 \log_2(10)$.
- RFC 4086 suggests password at least 29 bits of entropy to secure against **online attacks**.
- If cryptographic keys generated from password, **offline attacks** possible, password should have at least 128 bits of entropy.
- **Online vs. Offline attack:** Need to communicate with server not under control to check whether password is correct.
- E.g. WPA2 personal vulnerable to offline dict. attack. Some "hash" of password sent in clear.
- **Enhance password system:**
 - Make online attack difficult by adding intentional delay / lock.
 - Make offline attack difficult by applying KDF to password.
 - System check for weak password / regular changes of password.
- **Password vs. Secret Key:**
 - KDF: key derivation function.
 - Password as source for crypto secret key.
 - Using cryptographic hash, hash multiple times to form key, increase cost of exhaustive search. Tradeoff utility.

• Password Files:

Hashing:

- File stores userid, password. Add additional layer of protection.
- Password file should be "hashed" and "salted!"
- (Not "encrypted", don't want way to decrypt and get back)
- To authenticate, store and compare hash.

Salt:

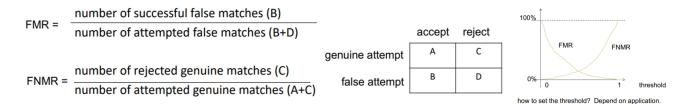
- Need same password hash to two diff. values, for two diff. userid.
- (Rainbow table: precomputed table caching outputs of crypto hash function to crack hashes)
- Achieve this by salting. New salt randomly generated, concat to pw., output different hash. Salt and hash stored in database.

ATM Skimming

- Demonstrates password stealing.
- Authentication: User presents card, and PIN.
- Data encoded into magnetic strip using well-known standards, easy to "copy" card by reading and writing to spoofed card.
- ATM Skimmer: card-reader, camera overlooking keypad / spoofed keypad, some means to record, transmit.
- **Measures:** Anti-skimmer devices, shielding keypad, increase awareness, change to unforgeable smartcard.

2.2 Biometric

- **Biometric data as password.** For identification, or verification.
- **Enrollment:** template of user biometric data captured, stored (bootstrapping). **Verification:** capture biometric data, compare using matching algorithm.
- Inevitable noise capturing biom. data, leading to error in matching.
 - **FMR (False Match Rate), FNMR (False non-match rate).**
 - Adjust threshold to adjust FMR & FNMR.
 - **EER (Equal Error Rate, FNMR = FMR)**
 - **FER (False-to-enroll rate, users' data uncaptureable, e.g. injury)**
 - **FTC (Failure to capture rate, uncaptureable during auth, e.g. finger dirty**



- **Fingerprint as biometric:** performance depends on quality of scanner, EER range from 0.5% to 5%.
- Biometric data easily spoofable, include *liveness detection* to verify entity.

2.3 Multi-factor / Multi-step Authentication

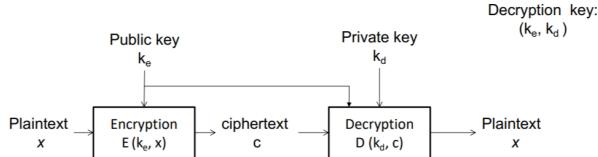
- **n-factor Authentication:** Require n authentication factors.
- **"Factors": Know (Pw, PIN), Have (card, phone, token), Are (Biometric).**
- Must be distinct factors for multi-factor (Know + Have, Have + Are) etc.
- One Time Password token: **Time-based /or Sequence-based**.
- **2-Step Verification:** Both same factor, e.g. (email + password, both "what-you-know", cannot be called "2-factor").
- **out-of-band:** In 2-step verification, two communication channels, main and separate for add. authentication. Non-main called "out-of-band" channel, assume attacker unable to compromise both channels.

3. Authenticity (Data Origin)

Crypto Primitive: Public Key Cryptography

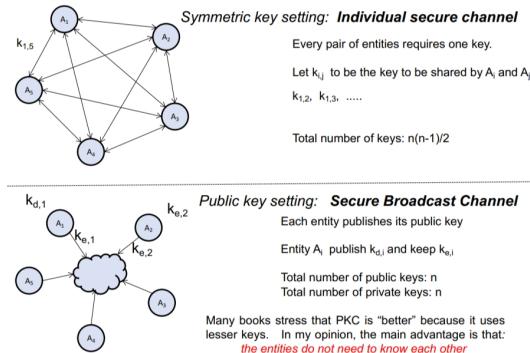
Public key crypto primitive includes public key encryption and signature. Goal is for confidentiality, and authenticity.

- Symmetric-key encryption uses same key for encrypt, decrypt.
- **Public Key (Asymmetric-key) scheme:** Uses different keys for encryption and decryption.
- Decryption algo also needs public key. When said "using private key to decrypt", this means using both.



- **Security Requirement:** Given public key & ciphertext, difficult to determine private key & plaintext. (Encryption oracle accessible.)
- **(PKC) Public Key Crypto:** Handles scenario of sharing symmetric key via secure channel. PKC only requires secure broadcast channel to distribute the public key. However, building secure broadcast channel to distribute public key is challenging.
- **PKC:** Useful for encryption, important application is in authentication (e.g. HTTPS). Limitations compared to symmetric in speed.

Keys Distribution



- **Remark:** Many PKC represent data as integers, algo used are some arithmetic operation on integers, represented using binary.
- A 1024-bit key is represented under binary representation. An algorithm exhaustively searches 1024-bit integers (2^{1024}) is infeasible.
- For this course, we will consider "**classroom**" RSA, most basic form RSA.
- **In Practice:** Padded RSA (to destroy some property), choose strong primes, fast secure ways to generate primes, secure implementation to guard against side-channel attack, "e" is fixed mostly at 65537. Considerations of quantum computer.

Popular PKC Schemes

RSA: Key size - 2048 bits.

ElGamal: Exploit techniques in Elliptic Curve Cryptography (ECC), reduce key size to - 300 bits.

Paillier: Partial homomorphic with respect to addition.

Public Key: RSA ("Classroom RSA")

"Classroom RSA" - setup

1. Owner randomly chooses 2 large primes p, q and computes $n = pq$.
 2. Owner randomly chooses an encryption exponent e s.t. $\gcd(e, (p-1)(q-1)) = 1$ (i.e. e does not have common factor with $(p-1)$ or $(q-1)$)
 3. Owner finds the decryption exponent d where $d \cdot e \text{ mod } (p-1)(q-1) = 1$
- There is an algorithm that finds d , when given e, p and q . We won't get into the details.
The term $(p-1)(q-1) = \Phi(n)$ is aka the Euler's totient function, which is the number of co-primes $< n$.
4. Owner publishes (n, e) as public key, and safe-keep (n, d) as the private key. (note that owner doesn't need to keep p, q)

Encryption, Decryption

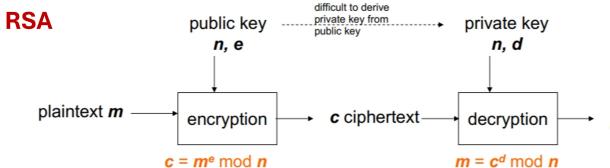
public key (n, e)	private key (n, d)
<ul style="list-style-type: none"> • Encryption: Given m, the ciphertext c is $c = m^e \text{ mod } n$ 	<ul style="list-style-type: none"> • Decryption Given c, the plaintext m is $m = c^d \text{ mod } n$

$$n = pq$$

$$\Phi(n) = (p-1)(q-1)$$

$$d = e^{-1} \text{ mod } \Phi(n)$$

RSA



Optional remark: Consider the relationship: $c = m^e \text{ mod } n$

- (RSA problem) Given c, n, e , find m , (called the e -th root of c)
- (discrete log problem): Given c, n, m , find e , (called the discrete log of c)

- **Correctness of RSA:** For any positive $m < n$, and any pair of public/private keys, that $\text{Decrypt}(\text{Encrypt}(m)) = m$.
- $(m^e)^d \text{ mod } n = m$
- Correctness depends on the property of modulo.

Example RSA

<p>There is an efficient algorithm that computes d from p, q, e. Details omitted.</p> <p>• Suppose $m = 9$</p> <p>Encrypt:</p> $c = m^e \text{ mod } n = 9^3 \text{ mod } 55 = 14$	<p>• Suppose $m = 9$</p> <p>Decrypt</p> $c^d \text{ mod } n = 14^{d,?} \text{ mod } 55$ <p>There is an efficient algorithm that computes d from p, q, e. Details omitted.</p> $= 14^{(p-1)(q-1)} \text{ mod } 55$ $= (36 \times 16 \times 49) \text{ mod } 55$ $= 9$
---	---

- **Interchangeable role of encryption & decryption key:** Swap role of d and e , public can decrypt, owner can encrypt. Usually does not hold in other PKC. This property useful in designing **signature scheme**.

- **Algorithmic Issues:** There is efficient algo to compute exponentiation, hence, efficient way to encrypt, and decrypt (given n, m, e , find $m^e \text{ mod } n$ for encrypt, given c, d, n , computer $c^d \text{ mod } n$).
- **Step 1 setup:** Find random prime, randomly pick number, test if prime. (Primality Test). **Step 3 setup:** Value of d efficiently computable from e and n using **extended Euclidean algorithm**.

Security of RSA

- Getting RSA private key from public key as difficult as factorizing n .
- However, not known whether the problem of getting plaintext from ciphertext as difficult as factorization.
- **post-Quantum cryptography:** Generally refers to PKC that are secure against quantum computer.
- Significant efforts to migrate current systems to post-quantum crypto (Quantum possibility to render RSA algo insecure by 2030). From 2016, NIST plans to choose standard by 2024. Four candidates for Rd 4 (2022).

Data Authenticity (Hash): Unkeyed Hash

Data Authenticity (Mac): Keyed Hash

Data Authenticity (Signature): Asymmetric Key

Attacks & Pitfalls

Birthday Attack on hash

Design flaw: Using encryption for authenticity

Time-Space tradeoff