Confidentiality	Asset only viewed by authorized parties
Integrity	Asset only modified by authorized parties
Availability	Asset available for use by authorized parties
Authenticity	Able to confirm identity of sender/origin of information Authenticity => Integrity
Public Key	Used by public to encrypt or verify MAC
Private Key	Kept secret, used to decrypt public key encrypted messages or to sign signature
Digital Signature	Public/Asymmetric-key hash. Sign with private key, verify with public key. Used to verify authenticity of message
MAC	Symmetric keyed hash, outputs tag t. Used to verify authenticity of message
Key Exchange	method in cryptography by which cryptographic keys are exchanged between two parties, allowing use of a cryptographic algorithm.
Station to Station	Authenticated key exchange based on Diffie Hellman by adding signature. In public-key cryptography, the Station-to-Station (STS) protocol is a cryptographic key agreement scheme. The protocol is based on classic Diffie–Hellman, and provides mutual key and entity authentication. Unlike the classic Diffie–Hellman, which is not secure against a man-in-the-middle attack, this protocol assumes that the parties have signature keys, which are used to sign messages, thereby providing security against man-in-the-middle attacks.
SSL/TLS	Sits on top of the Transport Layer. SSL/TLS first “protects” the data using encryption(for confidentiality) and MAC(for authenticity), and then instructs the transport layer to send the protected data An end-to-end encryption is performed
WPA2	WPA2 provides protection at layer 2 (link) and layer 1 (physical). A popular protocol employed in home WiFiaccess point
IPSec	IPsec provides integrity/authenticity protection of IP addresses, but not their confidentiality (in Internet Layer 3) Hence, attackers are unable to “spoof” the source IP address But they can still learn the source and destination IP addresses of the sniffed packets Vs

	Internet Protocol Security secures Internet Protocol communications by authenticating and encrypting each IP Packet of a communication session. End-to-end security scheme operating in Internet Layer
Buffer overflow	A data buffer (or just buffer): “a contiguous region of memory used to temporarily store data, while it is being moved from one place to another” E.g in call stack A buffer overflow, or buffer overrun, is an anomaly where a program, while writing data to a buffer, overruns the buffer's boundary and overwrites adjacent memory locations. E.g strcpy()
Integer overflow	An integer overflow occurs when an arithmetic operation attempts to create a numeric value that is outside of the range that can be represented with a given number of digits – either higher than the maximum or lower than the minimum representable value.
XSS	A type of injection attack on web apps, whereby a forum poster or web attacker attacks another web user by causing the latter run a (malicious) script from the former in the execution context of a page from an involved web server, thus subverting the Same Origin Policy In reflected XSS:the web server that returns a page reflecting the injected script In persistent XSS:the web server that stores a page containing the injected script
CSRF	Cross-Site Request Forgery (CSRF) is an attack that forces an end user to execute unwanted actions on a web application in which they’re currently authenticated. A type of authorization attack on web apps, whereby a web attacker attacks a web user by issuing a forged request to a vulnerable web server ‘on behalf’ of the victim user The attack disrupts the integrity of the victim user’s session This is, in a way, the reverse of XSS: it exploits the server’s trust of the client(the server believes that the issued request is from the client)
XSS vs CSRF	CSRF is already authenticated Cross-site scripting (or XSS) allows an attacker to execute arbitrary JavaScript within the browser of a victim user. Cross-site request forgery (or CSRF) allows an attacker to induce a victim user to perform actions that they do not intend to. The consequences of XSS vulnerabilities are generally more serious than for CSRF vulnerabilities: ● CSRF often only applies to a subset of actions that a user is able to perform. Many applications implement CSRF defenses in

	general but overlook one or two actions that are left exposed. Conversely, a successful XSS exploit can normally induce a user to perform any action that the user is able to perform, regardless of the functionality in which the vulnerability arises. ● CSRF can be described as a "one-way" vulnerability, in that while an attacker can induce the victim to issue an HTTP request, they cannot retrieve the response from that request. Conversely, XSS is "two-way", in that the attacker's injected script can issue arbitrary requests, read the responses, and exfiltrate data to an external domain of the attacker's choosing.
SQL Injection	SQL injection is a code injection technique used to attack data-driven applications, in which malicious SQL statements are inserted into an entry field for execution (e.g. to dump the database contents to the attacker). SQL injection must exploit a security vulnerability in an application's software, for example, when user input is either incorrectly filtered for string literal escape characters embedded in SQL statements or user input is not strongly typed and unexpectedly executed. SQL injection is mostly known as an attack vector for websites but can be used to attack any type of SQL database.
Clickjacking	Clickjacking is an attack that tricks a user into clicking a webpage element which is invisible or disguised as another element Clickjacking (classified as a user interface redress attack or UI redressing) is a malicious technique of tricking a user into clicking on something different from what the user perceives, thus potentially revealing confidential information or allowing others to take control of their computer while clicking on seemingly innocuous objects, including web pages.
Privilege escalation	Privilege escalation is the act of exploiting a bug, a design flaw, or a configuration oversight in an operating system or software application to gain elevated access to resources that are normally protected from an application or user.
Side Channel	In computer security, a side-channel attack is any attack based on information gained from the implementation of a computer system, rather than weaknesses in the implemented algorithm itself (e.g. cryptanalysis and software bugs). Timing information, power consumption, electromagnetic leaks or even sound can provide an extra source of information, which can be exploited.
Covert Channel	In computer security, a covert channel is a type of attack that creates a capability to transfer information objects between processes that are not supposed to be allowed to communicate by the computer security policy. Notice that a covert channel is a channel intentionally created by an attacker to leak information out of the target system; whereas a side channel is an unintentional channel taken advantage by an attacker to

	obtain more information about the target system. A side channel attack exploits a side channel.
Zero-Day	A zero-day (also known as 0-day) is a computer-software vulnerability either unknown to those who should be interested in its mitigation (including the vendor of the target software) or known and a patch has not been developed. Until the vulnerability is mitigated, hackers can exploit it to adversely affect programs, data, additional computers or a network.[1] An exploit directed at a zero-day is called a zero-day exploit, or zero-day attack.
Typo Squatting	Typosquatting is a type of social engineering attack which targets internet users who incorrectly type a URL into their web browser rather than using a search engine.
Click Fraud	Click fraud is the act of illegally clicking on pay-per-click (PPC) ads to increase site revenue or to exhaust a company's advertising budget.
Phishing	Phishing is a type of social engineering attack often used to steal user data, including login credentials and credit card numbers. It occurs when an attacker, masquerading as a trusted entity, dupes a victim into opening an email, instant message, or text message.
Pharming	Pharming is a cyberattack intended to redirect a website's traffic to another, fake site by installing a malicious program on the computer. Pharming can be conducted either by changing the hosts file on a victim's computer or by exploitation of a vulnerability in DNS server software.
Phishing vs Pharming	Phishing is a fraudulent practice where cybercriminals send you emails that appear to come from reputable organizations. The emails contain malicious links which take you to a fake website where unsuspecting users enter personal information – such as their username and password. Once you have submitted this information, fraudsters can use it for criminal purposes. Pharming is a form of phishing but without the enticement element involved. Pharming involves two stages: Firstly, the hackers install malicious code on your computer or server. Secondly, the code sends you to a fake website, where you may be deceived into providing personal information. Computer pharming doesn’t require that initial click to take you to a fraudulent website. Instead, you are redirected there automatically – where the pharmers then have access to any personal information you divulge. Phishing uses deceptive email, social media, or text messages asking you for your financial information, while pharming requires no lure. For this reason, pharming has been described as "phishing without a lure." Pharming is considered more dangerous than phishing since it can affect a significant number of computers without any conscious action from the victims. However, pharming attacks are less common than phishing because they require significantly more work from the attackers.

Fuzzing	Fuzz testing (fuzzing) is a quality assurance technique used to discover coding errors and security loopholes in software, operating systems or networks. It involves inputting massive amounts of random data, called fuzz, to the test subject in an attempt to make it crash.
Mandatory Access Control	a system-wide policy decides the rights, which must be followed by everyone in the system must follow
Discretionary Access Control	the owner of the object decides the rights
MAC vs DAC	With mandatory access control, this security policy is centrally controlled by a security policy administrator; users do not have the ability to override the policy and, for example, grant access to files that would otherwise be restricted. By contrast, discretionary access control (DAC), which also governs the ability of subjects to access objects, allows users the ability to make policy decisions and/or assign security attributes. (The traditional Unix system of users, groups, and read-write-execute permissions is an example of DAC.) MAC-enabled systems allow policy administrators to implement organization-wide security policies. Under MAC (and unlike DAC), users cannot override or modify this policy, either accidentally or intentionally. This allows security administrators to define a central policy that is guaranteed (in principle) to be enforced for all users. Historically and traditionally, MAC has been closely associated with multilevel security (MLS) and specialized military systems. In this context, MAC implies a high degree of rigor to satisfy the constraints of MLS systems. More recently, however, MAC has deviated out of the MLS niche and has started to become more mainstream. The more recent MAC implementations, such as SELinux and AppArmor for Linux and Mandatory Integrity Control for Windows, allow administrators to focus on issues such as network attacks and malware without the rigor or constraints of MLS.
Intermediate Access Control	Group subjects/objects and define access rights on defined groups We sometime use the term privilege to describe the access rights Privilege can also be viewed as an intermediate control
Role Based Access Control	Grouping determined by role of subject Subset of intermediate access control In RBAC systems, access is assigned by the system administrator and is stringently based on the subject's role within the household or organization and most privileges are based on the limitations defined by their job responsibilities. So, rather than assigning an individual as a security manager, the security manager position already has access control permissions assigned to it. RBAC makes life much easier

	because rather than assigning multiple individuals particular access, the system administrator only has to assign access to specific job titles.
Protection Rings	Each subject (e.g. process) and object (e.g. data) are assigned a number: The smaller the number is, the more important: we can call processes with lower ring number as having “higher privilege”; likewise an object with smaller number are more important Whether a subject can access an object is determined by the assigned numbers: If a process is assigned a number i: it runs in ring i A subject cannot access (read/write) an object with smaller ring number
Reference monitor	In operating systems architecture a reference monitor concept defines a set of design requirements on a reference validation mechanism, which enforces an access control policy over subjects' (e.g., processes and users) ability to perform operations (e.g., read and write) on objects (e.g., files and sockets) on a system. The properties of a reference monitor are captured by the acronym NEAT, which means: ● The reference validation mechanism must be Non-bypassable, so that an attacker cannot bypass the mechanism and violate the security policy. ● The reference validation mechanism must be Evaluable, i.e., amenable to analysis and tests, the completeness of which can be assured (verifiable). Without this property, the mechanism might be flawed in such a way that the security policy is not enforced. ● The reference validation mechanism must be Always invoked. Without this property, it is possible for the mechanism to not perform when intended, allowing an attacker to violate the security policy. ● The reference validation mechanism must be Tamper-proof. Without this property, an attacker can undermine the mechanism itself and hence violate the security policy.
Address Randomization	Address space layout randomization (ASLR) is a computer security technique involved in preventing exploitation of memory corruption vulnerabilities. In order to prevent an attacker from reliably jumping to, for example, a particular exploited function in memory, ASLR randomly arranges the address space positions of key data areas of a process, including the base of the executable and the positions of the stack, heap and libraries.

Same origin policy:

A script running in the browser could access cookies (and also webpage's objects)
Due to security concern, browser employs the following Same-Origin Policy access control
The script in a webpage A(identified by its URL) can access cookies stored by another webpage B(identified by its URL), only if both Aand Bhave the same origin
Origin is defined as the combination of protocol, hostname, and port number

Buffer Overflow:

A data buffer (or just buffer): "a contiguous region of memory used to temporarily store data, while it is being moved from one place to another"

In general, a buffer overflow refers to a situation where data is written beyond a buffer's boundary

A well-known function in C that is prone to buffer overflow is a string copying function: strcpy()

char s1[10]
strcpy(s1,s2)

In the above, the length of s2 can potentially be more than 10, since the length is determined by the first occurrence of null
The strcpy() may copy the whole string of s2 to s1, even if the length of s2 is more than 10
Since that the buffer size of s1 is only 10, the extra values will be overflowed and written to other part of the memory

Defense, use strncpy(s1, s2, n) instead and specify n = # characters copied
Also do bounds checking

Stack smashing: a special case of buffer overflow that targets a process' call stack
If the stack is being overflowed such that the return address is modified, the execution's control flow will be changed
A well-designed overflow could also "inject" the attacker's shellcode into the process' memory, and then execute the shellcode

Strings:

In C, printf()adopts an efficient representation:
The length is not explicitly stored
The first occurrence of the null character \0 (i.e. byte with value 0) indicates the end of the string,thus implicitly giving the length

In general: a blacklisting-based filtering could be incomplete due to the “flexible use” of character encoding
Whitelist better than blacklist

Ip address vulnerability => integers can be negative so won’t be in blacklist but generate same ip address

Never trust the input from user
Always convert them to a standard (i.e. canonical) representation immediately
Preferably, do not rely on the verification check done in the application; i.e. do not rely on the application developers to write the verification
Rather, try to make use of the underlying system access control mechanism

A backdoor: a covert method of bypassing normal authentication
Such access points are also known as Easter eggs

Defense against buffer overflow:

Use safer functions like strncpy()

Input validation
White list: A list of items that are known to be benign and allowed to pass, which could be expressed using regular expression However, some legitimate inputs may be blocked.
Black list: A list of items that are known to be bad and to be rejected. For example, to prevent SQL injection, if the input contains meta characters, reject it. However, some malicious input may be passed.
A filter that completely blocks all bad inputs and accepts all legitimate inputs is very difficult to design, whitelist better

Bounds and type checking
Check array upper and lower bounds manually since c, c++ do not check

Memory Protection
Canaries are “secret” values inserted at carefully selected memory locations at runtime
Checks are carried out at runtime to make sure that the values are not being modified: if so, halts
Canaries can help detect overflow, especially stack overflow:

In a typical buffer overflow, consecutive memory locations have to be over-ran: the canaries would be modified
It is important to keep the values “secret”:if the attacker knows the value, it may be able to write the secret value to the canary while over-running it!

Address space layout randomization (ASLR) is a prevention technique that can help decrease the attacker’s chance
ASLR: randomly arranges the address space positions of key data areas of a process, including: the base of the executable and the positions of the stack, heap and libraries

Apply the “Principle of Least Privilege”:
When writing a program, be conservative in elevating the privilege
When deploying software system, do not give the users more access rights than necessary, and do not activate unnecessary options

Cryptology = Cryptography + Cryptanalysis

The National Security Agency (NSA): a national-level intelligence agency of the US Dept of Defense, which is responsible for global monitoring, collection, and processing of information and data for foreign intelligence and counterintelligence purposes, specializing in a discipline known as signals intelligence (SIGINT).

The National Institute of Standards and Technology(NIST): a measurement standards laboratory, and a non-regulatory agency of the US Dept of Commerce, whose mission is to promote innovation and industrial competitiveness.

Cryptography backdoor: a method, often secret, of bypassing normal encryption in a cryptosystem. It allows an intruder to access the plaintext without having the correct user credentials.

Key escrow: an arrangement in which the keys needed to decrypt encrypted data are held in escrow so that, under certain circumstances, an authorized third party may gain access to those keys.

Decryption order: an order that forces suspects to decrypt their encrypted data or give up their keys.

Whitfield Diffie: one of the pioneers of public-key cryptography; co-inventor of Diffie-Hellman key exchange; won the 2015 Turing Award.

Ron Rivest: co-inventor of the RSA algorithm; inventor of the symmetric-key encryption algorithms RC2/RC4/RC5; inventor of the MD2/MD4/MD5/MD6 cryptographic hash functions; co-author of \Introduction to Algorithms" book; won the 2002 Turing Award.

Alice, Bob, Eve, Mallory: Check https://en.wikipedia.org/wiki/Alice_and_Bob#Cast_of_characters.

Trent (or Ted): a trusted arbitrator as a neutral third party.

Graphical passwords: A graphical password or graphical user authentication is a form of authentication using images rather than letters, digits, or special characters. The type of images used and the ways in which users interact with them vary between implementations.

Covert channel: In computer security, a covert channel is a type of attack that creates a capability to transfer information objects between processes that are not supposed to be allowed to communicate by the computer security policy.

Side Channel Attack: In computer security, a side-channel attack is any attack based on information gained from the implementation of a computer system, rather than weaknesses in the implemented algorithm itself (e.g. cryptanalysis and software bugs). Timing information, power consumption, electromagnetic leaks or even sound can provide an extra source of information, which can be exploited.

Notice that a covert channel is a channel intentionally created by an attacker to leak information out of the target system; whereas a side channel is an unintentional channel taken advantage by an attacker to obtain more information about the target system. A side channel attack exploits a side channel.

End-to-end encryption: End-to-end encryption (E2EE) is a system of communication where only the communicating users can read the messages. In principle, it prevents potential eavesdroppers – including telecom providers, Internet providers, and even the provider of the communication service – from being able to access the cryptographic keys needed to decrypt the conversation
End-to-end encryption is intended to prevent data being read or secretly modified, other than by the true sender and recipient(s). The messages are encrypted by the sender but the third party does not have a means to decrypt them, and stores them encrypted. The recipients retrieve the encrypted data and decrypt it themselves.

Because no third parties can decipher the data being communicated or stored, for example, companies that provide end-to-end encryption are unable to hand over texts of their customers' messages to the authorities.

Single Sign On (SSO): Single sign-on (SSO) is an authentication scheme that allows a user to log in with a single ID and password to any of several related, yet independent, software systems. True single sign-on allows the user to log in once and access services without re-entering authentication factors.

Hardware RNG: In computing, a hardware random number generator (HRNG) or true random number generator (TRNG) is a device that generates random numbers from a physical process, rather than by means of an algorithm. Such devices are often based on microscopic phenomena that generate low-level, statistically random "noise" signals, such as thermal noise, the photoelectric effect, involving a beam splitter, and other quantum phenomena. These stochastic processes are, in theory, completely unpredictable for as long as an equation governing such phenomena is unknown or uncomputable, and the theory's assertions of unpredictability are subject to experimental test. This is in contrast to the paradigm of pseudo-random number generation commonly implemented in computer programs.

Quantum RNG: Quantum RNGs exploit elementary quantum optic processes that are fundamentally probabilistic to produce true randomness. As the quantum processes underlying the QRNG are well understood and characterized, their inner workings can be clearly modelized and controlled to always produce unpredictable randomness.

Retinal vs Iris Scan: Retina scans are 70x more accurate than iris scans. Iris scans capture an image of the iris from a distance, while retina scanning does it by placing the person’s eye near to an eyepiece. Retina scanning is best suited for physical identification

Be aware of the differences between pseudo random number generators and hardware random number generators by reading, for instance https://en.wikipedia.org/wiki/List_of_random_number_generators.

Nonce: In cryptography, a nonce (number once) is an arbitrary number that can be used just once in a cryptographic communication. It is often a random or pseudo-random number issued in an authentication protocol to ensure that old communications cannot be reused in replay attacks. They can also be useful as initialization vectors and in cryptographic hash functions.

Encryption

Correctness m = Dk(Ek(m)) Ek, Dk must be one to one function
Efficiency encrypt, decrypt, generate key must be fast
Security difficult to recover secret key or plaintext
Computational security => 2^128 keys

Attacks on Cipher

ciphertext only – a large number of ciphertexts all encrypted using the same key
known plaintext – pairs of ciphertext and the corresponding plaintext

Substitution Cipher

Substitution table representing 1-1 mapping from U to U
Monoalphabetic (1-1 mapping/ substitution fixed for each alphabet)
Key space => set of all possible keys
Key space size = |U|! E.g 27! For alphabet + '_'
Key size => minimum number of bits to represent all possible keys
Key size = log2(key space size) e.g log2(27!) = 94 bits = log(27!)/log(2)

Encryption = mapping, Decryption = inverse

Known plaintext

1. Brute force => check all possible keys to find key where Ek(X) = C
    a. (27!) loops/operations
2. Determine key given plaintext and ciphertext just by matching
    a. Sufficiently enough or long ciphertext/plaintext pair can determine full table

Insecure under known plaintext because of 2.
Ciphertext only

3. Brute force => check all possible keys to find key where Dk(C) = english/semantic
    a. (27!) loops/operations
    b. Small probability wrong key
4. Frequency analysis on monoalphabetic cipher
    a. Compare frequency with letter frequency distribution
    b. Map frequently occurring letters in ciphertext with frequently occurring letters in english

Insecure under known ciphertext because of 4.

Shift/Caesar Cipher

Each letter in plaintext shifted down fixed number of places e.g rot13
Monoalphabetic
Key space size = |U| e.g 27 (alphabet + '_')
Key size = log2(|U|)

Easy to break with brute force
Same attacks as Substitution (subset of substitution ciphers)

Vignere Cipher

Shift by keyword(string of letters representing numbers based on position in alphabet)
Keyword repeated for longer plaintext
Polyalphabetic (each character shift based on current index of keyword)
Same character can be mapped to different characters

Each character of keyword has 27 choices (alphabet + space/underscore)
Key space size = 27^|length of key|
Key size = log2(27^|length of key|)

Vignere > substitution
27^x > 27!
Length of keyword at least 20

Insecure under known plaintext => determine key by distance of each plaintext character and ciphertext character, key repeats for longer plaintext

Insecure against ciphertext only given we can determine length/period of keyword
All letters whose index is i(mod k) gets shifter by same character in keyword (monoalphabetic)
Frequency analysis on each group shifted by same key

Determine period of keyword with kasiski method, repeated patterns

Permutation Cipher

Permutation to each block
Key space size = |length of block|!
Key size = log2(|length of block|!)

Insecure under known plaintext, just match
Insecure under ciphertext only if plaintext in English

One time Pad

Key as long as plaintext
Encrypt => plaintext XOR key to get ciphertext
Decrypt => ciphertext XOR key to get plaintext

Insecure under known plaintext but key will not be reused
One time pad leaks no information of plaintext except length
Perfect secrecy/unbreakable

If key k is random, ciphertext c looks as random as key

If k[0] = 0, X[0] = C[0]
If k[0] = 1, X[0] = C[0]'

Both probability = ½

Fails with repeated key
Key must be as long as plaintext

Attacker Capabilities (weakest to strongest)

Ciphertext only same key
Known plaintext => can observe ciphertext and know corresponding plaintext, same key
Chosen plaintext => can encrypt plaintext of choice, same key
Chosen ciphertext => can decrypt ciphertext of choice + encrypt plaintext, same key

Side channel attack => exploit source of info that depends on implementation of cipher
Observe/measure analog characteristics but cannot alter integrity
E.g execution time, power, noise

Invasive attack => can alter integrity

Modern Cipher

Block cipher high diffusion => change to key/plaintext spread across ciphertext
Stream cipher low diffusion

Confusion => complex transformation, unable to predict change

Substitution => confusion
Permutation => diffusion

Sniffing is the process in which all the data packets passing in the network are monitored. ... Sniffers can be hardware or software installed on the system. Spoofing is the process in which an intruder introduces fake traffic and pretends to be someone else

Boiled down: phishing aims to take hold of personal information by convincing the user to provide it directly; spoofing aims to steal or disguise an identity so malicious activity can ensue. Both employ a level of disguise and misrepresentation, so it is easy to see why they are so closely paired.

Skimming in cybersecurity refers to cybercriminals' strategies for capturing and stealing cardholder's personal payment information. Identity thieves use various approaches to obtain card data. One of the most advanced methods is using a small skimming device designed to read a credit card's microchip or magnetic strip information. Criminals can execute skimming attacks whenever a cardholder opts for electronic payment methods in a physical location.

Skimming => card reader + camera for PIN

False Match Rate = successful false match/all false attempts = fake but get in

False Non Match Rate = rejected genuine attempt /all genuine = real but rejected

Increase Threshold = stricter = reduce FMR = increase FNMR

0 Threshold = anyone can get in
1 Threshold = nobody can get in

Failure to enroll => cannot register biometrics
Failure to capture => fail to capture during authentication

2FA => 2 factors for authentication

E.g password, OTP, biometrics

SMS OTP can be intercepted if message not end to end encrypted

Correctness: D(kd , E(ke ,m)) = m
Trapdoor function E():
E()can be efficiently evaluated using the publicly-known ke
But its inverse D() is computationally infeasible without the corresponding private key kd(as the trapdoor information)
That is, without the trapdoor kd, there's an asymmetric computational requirement between E() and its inverse D()
Security: For all ke and all c, the plaintext m corresponding to c can be recovered only with the kd corresponding to ke

In other words, the security requirement is: given the public key and the ciphertext, but not the private key, it is difficult to determine the plaintext

(A more refined formulation of the requirement:without a knowledge of the private key, the ciphertext resembles a sequence of random values)

The requirement implies that: it must be difficult to get the private key from the public key

Furthermore, RSA has some interesting properties, e.g. homomorphic property:
$E(m1 * m2)$ can be directly evaluated by using E(m1)and E(m2) without decrypting $m1 * m2$

Not 1 way => not collision resistant

Collision resistant => 1 way

If h(F) = h(F') , very high confidence F = F' since collision resistant

Work factor on breaking cipher is M to find collision with high probability

$M > 1.17T^{0.5}$

Digest length must be higher than key length

Suppose Alice's certificate is issued & signed by CA1, which is a tier-1 intermediate CA
Alice must include her email, her certificate, and CA1 certificate
Now, Bob can:
Verify CA1's certificate: using root CA's public key
Verify Alice's certificate: using the verified CA1's public key
Verify Alice's email: using Alice's verified public key

- Illustration:

PKI = Certificate + CA + trust hierarchy + certificate revocation

Typosquatting => just wait for people to click spoofed link

Strong Authentication
Cannot be exploited by replay attack from eavesdropping

Secret key challenge response
A hello B
B randomly pick m, send Ek(m) (m random to ensure freshness/no replay attack)
A decrypt Ek(m) send m to B
B verify and authenticates A (unilateral authentication from B to A)

Public key challenge response
A hello B
B choose random number r send to A
A use private key to sign r, send sign(r), certificate
B verify certificate, get public key, verify sign(r)

A authenticate B => uses B's private and public keys
A sends the nonce
B signs using private key
A verify (need know B public key, B sends certificate at start of authentication protocol)

If M can intercept after strong authentication, still can perform MITM

Outcome of strong authentication must be a new session keys (one for encrypt, one for authentication)

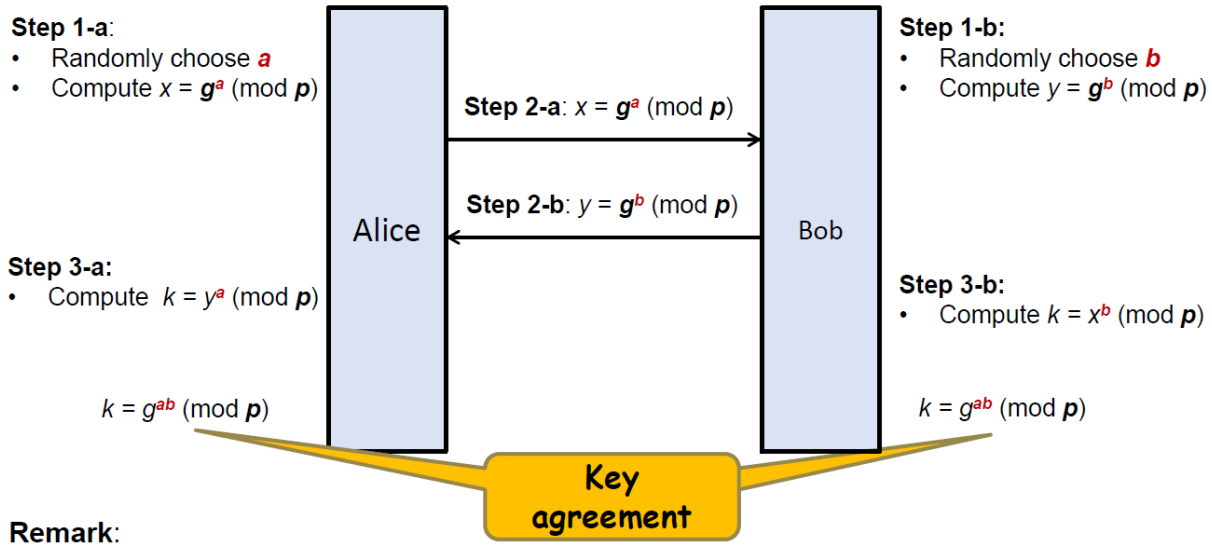
The process of establishing a secret between Alice & Bob is called key exchange, key establishment,or key agreement
Subsequent communication between Alice & Bob must be secured using the session key

Key exchange protocol must not leak info

Public key unauthenticated key exchange:
Alice generates a pair of private/public key
Alice sends the public key keto Bob
Bob carries out the following:
Randomly choose a secret k
Encrypt k using ke
Send the ciphertext c to Alice
Alice uses her private key kdto decrypt and obtain k.

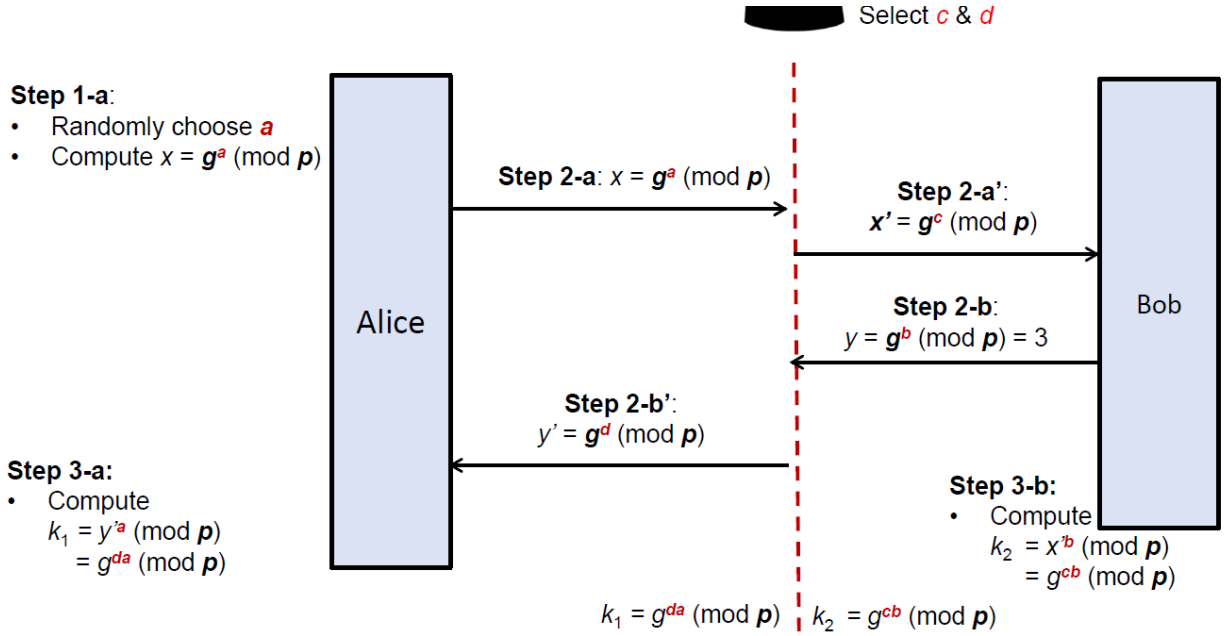
Diffie Hellman

- We assume both Alice and Bob have agreed on two ***publicly-known*** parameters: a ***generator g,*** and a large (e.g. 1,000-bit) prime ***p***



Remark:
Steps 1-a & 1-b, Steps 2-a & 2-b, and Steps 3-a & 3-b can be carried out ***in parallel***

Unauthenticated key exchange insecure in presence of M



From the attack: DH does ***not*** achieve entity authentication

A and M establish k1
M and B establish k2
M can see and modify message between A and B

Authenticated key exchange, just use key exchange but sign all communications

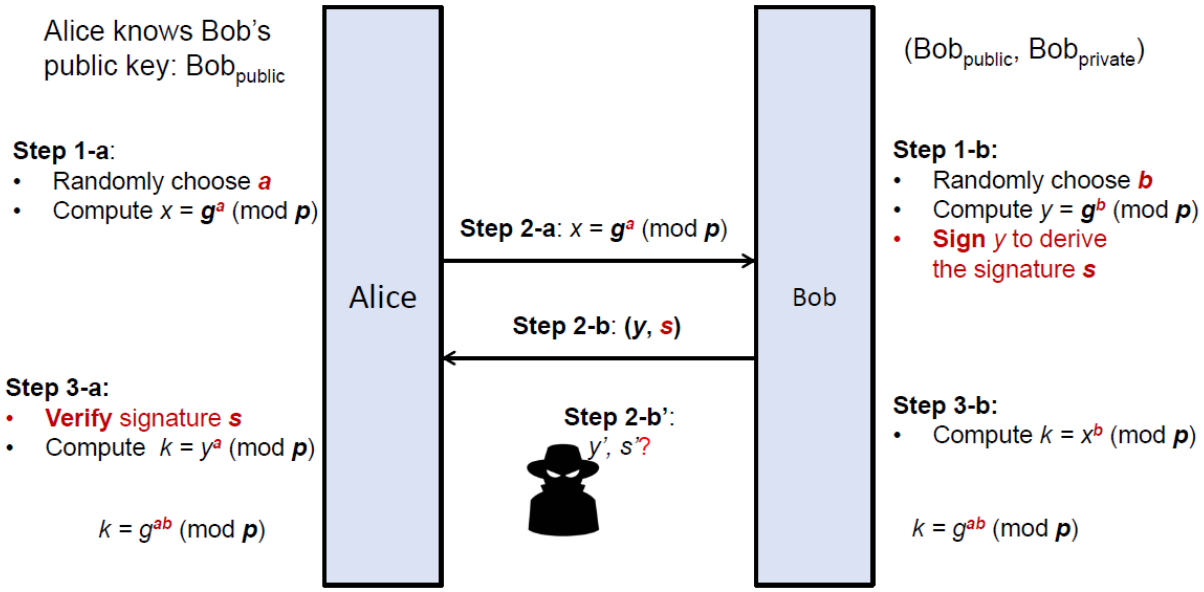
Secure channel between Alice browser(client) and web server(server)

Authenticated key exchange produces session keys k, t
Confidentiality protected using encryption with k
Authenticity protected using MAC with t

Subsequent communication between Alice and Bob.com will be protected by k, t and a sequence number i
Suppose m1, m2, m3, … are the sequence of message exchanged, the actual data to be sent for mi will be:
Ek(i||mi) ||MACt(Ek(i||mi))
where: i is the sequence number,

Station-to-Station (STS) Protocol

- Here, we consider unilateral authentication: Alice want to authenticate Bob
- Adding **signature** to DH: **authenticated** key-exchange based on DH



How about the unsigned x in Step 2-a?

HTTPS built on top of TLS/SSL between transport and application layers
=HTTP + SSL/TLS
How does TLS work at the high level?
Ciphers negotiation
Authenticated Key Exchange: the exchange of session key, which also authenticates the identities of parties involved
Symmetric-key based secure communication
Re-negotiation(if needed)

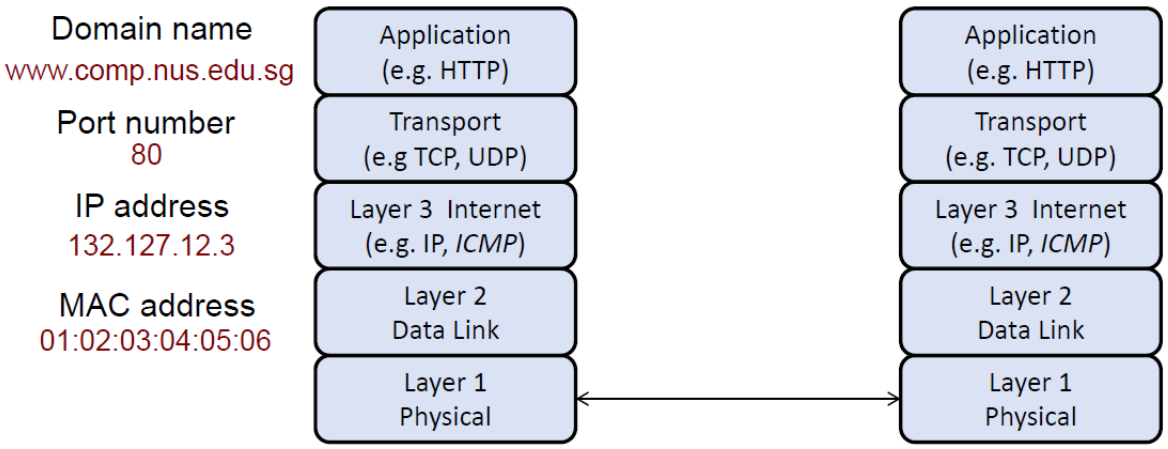
Authenticated encryption
symmetric encryption that returns both ciphertext and authentication tag
Authenticated encryption process: AE(KAB, M) = (C, T)
Decryption process: AD(KAB, C, T) = M only if T is valid

MAC then encrypt
SSL/TLS
Issue: a decryption is still needed on a corrupted message

Encrypt then MAC
IPSec
Feature: a decryption is not performed on a corrupted message

Authenticated cipher
AES-GCM TLS

- Different ***addressing schemes*** at different layers



A message to be sent is called: layer-N protocol data unit (PDU)
Encapsulation of upper (i.e.N+1) layer's PDU

UDP connectionless
TCP/IP is connection oriented
TCP is reliable: it has mechanisms to re-transmit, re-order, acknowledge packets so that the destination can receive the sent messages in the correct order

Resolution protocols
Domain Name System (DNS): (nslookup)
Maps domain name to IP address
A hierarchical decentralized naming system
An attacker can target the association of domain name with IP address
In this module, we only consider a basic type of DNS attack
DNS operates at the application layer

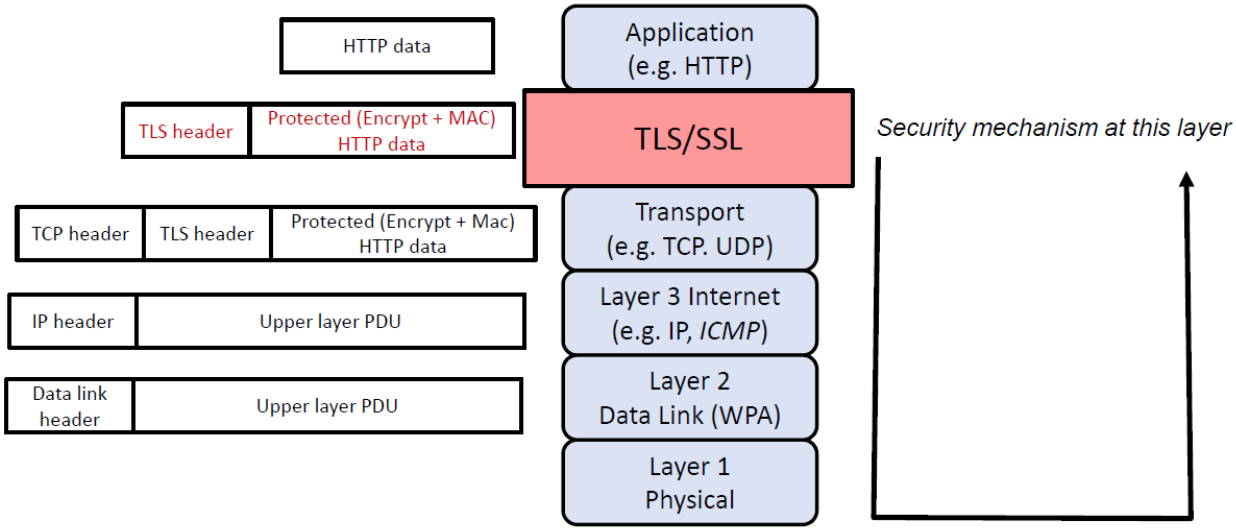
Address Resolution Protocol (ARP):
Associate/map IP address (logical address) with/to MAC address (physical address)
Use a broadcast mechanism on a local network
An attacker on the local network can target the association

Nmap: port scanner to determine which ports are open/listening
When a server receives an incoming packet, it will decide which application process to handle that packet based on the port no
By saying that a process/service is “listening” to a particular port, we mean that the process is running and ready to process packets with that particular portno

Wireshark: network packet capture and analyser

Security protocol that protects layer k would protect info from that layer and above against attacker at layer k-1 and below

SSL/TLS sit on top of transport layer
We can imagine that, when an application (e.g.browser or email agent) wants to send data to the other end point, it first pass the data and the destination IP address to SSL/TLS
Next, SSL/TLS first “protects” the data using encryption(for confidentiality) and MAC(for authenticity), and then instructs the transport layer to send the protected data
An end-to-end encryption is performed



The receiver end-point **decrypts** the received data at the corresponding layer

Attacker at physical layer
Can the attacker learn:
- Alice's uploaded report?
No, as it is protected by SSL/TLS, which protects the application layer, and information from that layer would be protected from an attacker below it, i.e. physical layer.
- The fact that Alice is visiting the LumiNUS website i.e. can the attacker lean the website's IP address?
Yes, as the information is contained in the IP headers from the network layer, which is not protected by TLS/SSL that sits on top of the transport layer from the attacker below it at the data link or physical layer.

Attacker at application layer
Alice's report?
Yes, as the SSL/TLS does not protect information from its layer and above from attackers who are also on its layer or above. The malware in the application is still able to sniff and spoof.
- Alice's MAC address?
No, as the MAC address is defined at layer 2. An attacker at the application layer is unable to attack "downwards", and can only attack upwards.

WPA2 provides protection at layer 2 (link) and layer 1 (physical)

Attacker at physical layer
Alice's report?
No, as data from the upper layers have been encrypted with AES.
- The fact that Alice is visiting LumiNUS website?
No
- The MAC address (which is assigned in the link layer)?
The MAC address is never encrypted, as the MAC itself is required to enable the packet to reach the router and enable the router to send packets back.

IPsec provides integrity/authenticity protection of IP addresses, but not their confidentiality:

Firewall
There is a need to control the flow of traffic between networks, especially between the untrusted public network (Internet) and the trusted internal network
Even within the internal network, we still need to divide it into different network segments and deny unnecessary access: Principle of least privilege, compartmentalization

Firewall sits at the border between networks
Looks at ip address, service and other characteristics
Controls ingress and egress filtering

Demilitarized Zone (DMZ):
A small sub-network that exposes the organization's external service to the (untrusted) Internet
The original military term: an area between states in which military operations are not permitted

Firewall rules
Examples of rules for Firewall2 (back-end firewall):
Block HTTP
Allow from Internal network to Mail Server: SMTP, POP3

Examples of rules for Firewall1 (front-end firewall):
Allow from anywhere to Mail Server: SMTP only

The table is processed in a top-down manner
The first matching rule determines the action taken
Hence: put your most specific rule first, and put your most general rule last

No	*Source IP*	*Dest IP*	*Source port*	*Dest port*	*Protocol*	*Direction*	*Action*
1	`Web-server`	*	`HTTP`	*	`TCP`	`OUT`	`Allow`
2	*	`Web-server`	*	`HTTP`	`TCP`	`IN`	`Allow`
3	*	*	*	*	*	*	`Block`

allow packets to be sent from (into) the `Web-server` to (from) any other IP addresses, and block everything else. Note that the symbol "*" means "any", and it matches anything.

The access control model/system gives a way to specify & enforce restriction of operations on objects by principals/subjects

A principal(or subject) wants to access an object with some operation
The reference monitor either grants or denies the access

UNIX is DAC and ACL

Access Control List => access rights of objects (objects as key)
Capabilities => subject is key

Intermediate control => groups

Owner, group, others

Root is superuser account in unix UID 0 username root

Unix has 2 rings => Superuser and normal/non root

/etc/password => user database and password file
/etc/shadow => hashed user passwords
/etc/group => group database

Principals => UID, GID
Subjects => processes invoked by an executable file PID
file(-)
directory(d)

- Octal mode notation:
  - This notation contains 3 or 4 octal digits.
  - The 3 rightmost digits refer to the permissions for the file user, the group, and others respectively.
  - There is an optional leading digit, allowing 4 digits to be given. This digit specifies the special file permissions:
    - setuid
    - setgid
    - sticky bit

#	Permission	rwx	Binary
7	read, write and execute	rwx	111
6	read and write	rw-	110
5	read and execute	r-x	101
4	read only	r--	100
3	write and execute	-wx	011
2	write only	-w-	010
1	execute only	--x	001
0	none	---	000

#	Permission	rwx	Binary
4	setuid	s/S	100
2	setgid	s/S	010
1	sticky	t/T	001
0	none	-	000

S bit elevates to file owner, has to be root for privilege escalation
The “s” (set-UID) bit indicates that the privilege is elevated to the file owner(in this case root) while a user is running this process:
To be able to change effective uid, must first be able to execute program with setuid bit
So program to change uid must be world executable --s--x--x

A process is also associated with process credentials: a real UID and an effective UID

When a process wants to access a file, the effective UID of the process is treated as the “subject” and checked against the file permission to decide whether it will be granted or denied access

Blacklisting based filtering incomplete due to flexible use/substitution of characters

Just as the name suggests, whitelisting is the opposite of blacklisting, where a list of trusted entities such as applications and websites are created and exclusively allowed to function in

the network. Whitelisting takes more of a trust-centric approach and is considered to be more secure

Fuzzing sends malformed inputs to discover vulnerabilities during testing

In operating systems architecture a **reference monitor** concept defines a set of design requirements on a reference validation mechanism, which enforces an access control policy over subjects' (e.g., processes and users) ability to perform operations (e.g., read and write) on objects (e.g., files and sockets) on a system. The properties of a reference monitor are captured by the acronym NEAT, which means:

- The reference validation mechanism must be *Non-bypassable*, so that an attacker cannot bypass the mechanism and violate the security policy.
- The reference validation mechanism must be *Evaluable*, i.e., amenable to analysis and tests, the completeness of which can be assured (verifiable). Without this property, the mechanism might be flawed in such a way that the security policy is not enforced.
- The reference validation mechanism must be *Always invoked*. Without this property, it is possible for the mechanism to not perform when intended, allowing an attacker to violate the security policy.
- The reference validation mechanism must be *Tamper-proof*. Without this property, an attacker can undermine the mechanism itself and hence violate the security policy.

With mandatory access control, this security policy is centrally controlled by a security policy administrator; users do not have the ability to override the policy and, for example, grant access to files that would otherwise be restricted. By contrast, discretionary access control (DAC), which also governs the ability of subjects to access objects, allows users the ability to make policy decisions and/or assign security attributes. (The traditional Unix system of users, groups, and read-write-execute permissions is an example of DAC.) MAC-enabled systems allow policy administrators to implement organization-wide security policies. Under MAC (and unlike DAC), users cannot override or modify this policy, either accidentally or intentionally. This allows security administrators to define a central policy that is guaranteed (in principle) to be enforced for all users.

Historically and traditionally, MAC has been closely associated with multilevel security (MLS) and specialized military systems. In this context, MAC implies a high degree of rigor to satisfy the constraints of MLS systems. More recently, however, MAC has deviated out of the MLS niche and has started to become more mainstream. The more recent MAC implementations, such as SELinux and AppArmor for Linux and Mandatory Integrity Control for Windows, allow administrators to focus on issues such as network attacks and malware without the rigor or constraints of MLS.

CSRF => user already authenticated

Cross-site scripting (or XSS) allows an attacker to execute arbitrary JavaScript within the browser of a victim user.
Cross-site request forgery (or CSRF) allows an attacker to induce a victim user to perform actions that they do not intend to.

Privilege escalation is the **act of exploiting a bug**, a design flaw, or a configuration oversight in an operating system or software application to gain elevated access to resources that are normally protected from an application or user.