INTRODUCTION TO COMPUTER AND NETWORK SECURITY

1. In the context of cryptography, (a) state and describe the Kerckhoffs principle, (b) explain the purpose of key management, (c) explain the notion of symmetric key cryptography, and (d) explain the notion of asymmetric (or public) key cryptography.
   (a) **Kerckhoffs principle**: it adfirms that we shouldn't rely on the secrecy of the encryption algorithm, but on the secrecy of the cryptographic key. In fact we should choose the best known algorithm, so that many people can work on it and improve it, but the key used for every communication should remain secret. So now, our problem is about key management.
   (b) **Key management**: the main purpose of key management is to ensure that the secret key used in an encrypted communication is known only by the parties that partecipate on it. To ensure it key should be as random as possible, so they need to have high entropy (high variance). Moreover key management is not only about key sharing but also, how key are stored, generated and destroyed.
   (c) **Symmetric key cryptography**: is the use of a single shared secret to share encrypted data between parties. Ciphers in this category are called symmetric because you use the same key to encrypt and to decrypt the data, hence using a cryptosystem writing: D(k,E(k,p))= p.
   Two types of symmetric key encryption exist: stream cipher (*encrypt sequences of short data blocks using a changing key stream, encryption is usually performed with XOR, security relies on design of key stream generator)* and block cipher (*encrypt sequences of long data blocks using a fixed key stream, encryption is usually performed using transposition and substitution, security relies on design of encryption function)*.
   d) **Asymmetric (or public) key cryptography**: also called PKC, it's based on the use of a public key, known by every user and shared along the communication channel, and a private key, known only by the owner and never shared, this ensure a secure communication even if the channel is not secure. The two keys are mathematically related but the use of one doesn't allow to determine the other, this can be done thanks to the use of 1-way trapdoor function (such as multiplication vs factorization). PKC is computationally expensive so one solution is to use it only when a symmetric key is being changed.
2. In the context of the Diffie-Hellman key exchange, describe (a) the protocol, (b) the Men-In-the- Middle (MITM) attack, and (c) how it is possible to mitigate the MITM attack.
   (a)**DH protocol**: this protocol is used only for key exchange, not for authentication an digital signature. The protocol is based on the fact that computation of exponential are easier than that related to logarithms, so we can consider it a one way function. The main step of DH protocol are the following(we consider p as a prime and g as a generator):

   - Alice chooses a large number “a”
   - Compute A=g^a(mod p)
   - A is the public key and so it's shared with Bob, so Bob can compute his K.
   - The secret key that will be used in order to encrypt future messages with symmetric cryptography is K=B^a(mod p) which is also equal to A^b(mod p)

   - Bob chooses a large number “b”
   - Compute B=g^b(mod p)
   - B is the public key and so it's shared with Alice, so Alice can compute her K
   - The secret key that will be used in order to encrypt future messages with symmetric cryptography is K=A^b(mod p) which is also equal to B^a(mod p)

   So K=B^a(mod p)=A^b(mod p) is the secret key exchanged and generated using a PKC algorithm. Moreover the parameters g and p are shared.
   (b)**MITM attack**: a MITM attack happens when in a communication there's a third party who is not authorized to partecipate on it. This party usually alters the communication between the two parties that believe that thei conversation is private so they share sensitive information. A middle man hence can impersonates one of the two parties, since DH protocol doesn't provide authentication, and the parties send data that are intercepted by the man in the middle. Also called eavesdropping attack since information are sniffed by a third subject that doen't partecipate directly in the communication.
   (c)How mitigate it? One way to prevent MITM attack is finding a way to authentically bound the public key to the identity of the party which controls the corresponding private key, as this scope we need digital certificate, they ensure that we are talking with we are intended to do.

3. Define the notions of vulnerability and threat and give (at least) an example for each one.

Vulnerability: is a **weakness** inside the information system that can be exploited by a threat source. Some examples of vulnerability are: hidden backdoors, bugs in some software and the most common one is the use of weak password used by the user.
Threat: is an **activity**, that can be intentional or not, with th potential for causing harm to a system or an activity, a threat usually exploits a vulnerability. Some common threats are hackers, viruses and worms, which are self-replicating programs that don't require the action of a user in order to activate themselves.They usually exploit a weakness in a network to spread themselves.

4. Define the notions of risk and vulnerability and give (at least) an example for each one.
Vulnerability: is a **weakness** inside the information system that can be exploited by a threat source. Some examples of vulnerability are: hidden backdoors, bugs in some software and the most common one is the use of weak password used by the user.
Risk: it's the probability that a particular vulnerability will be exploited by a threat. A risk is proportional to the likelihood that a vulnerability can be exploited and also to the impact that this can have. We need to underline that this depends also on the different stackholders involved in a risk (for example risks are different for an user and for a worker in a bank system). We can summarize that risk is equal to likelihood*impact

5. In the context of TLS and the PKI, describe: (a) what a digital certificate is and what are its main components [10 points]; (b) what a root CA is [5 points]; (c) how a system can validate a certificate [5 points]; (d) how TLS 1.2 handshake works by drawing the message sequence chart and briefly describing each step [25 points]; (e) how TLS ensures authentication, confidentiality, and integrity [5 points].
(a)**Digital certificate and main components**: it's an arrangement that binds public keys with respective identities of entities, people and organizations. A digital certificate contains identity and public key, and it is signed by a Trusted Third Party, also called Certificate Authority (CA). So the main parts of a digital certificate are: Issuer (CA), Subject, Subject public key, Issuer digital signature and validiity period.
(b)**Root CA**: it's the entity that attests the correctness of the binding between identity and the public key, of course also CA is need to be trusted, so we adopt the “chain of trust” and to avoid an infinite regression, at the end of the chain we find the root trusted party, which is usually harcoded in the browser, and has a *self-signed certificate*.
(c)**How system can validate a certificate**: a certificate has a limited time of validity, so the Validation Authority (VA) has the role to check time validity and also checks signatures and also checks that the required certificate is not in CRL (to check revocation status). Also certificate chain is checked.
(d)**TLS handshake**:TLS is made up of two main protocol: handshake and record protocol. The first one is used to share a secret key (using always Diffie-Helmann), establish which versione of TLS will be used and which cipher will be used. Here the public key of each parties is used.
    - Client hello: contains the version of the protocol that the client wants to use and the list of supported cipher suites. Can also include a MAC authentication code to ensure integrity and authentication of messages.
    - Server hello: contains chosen protocol, cipher suite and session id. Moreover if a client requested an authentication then a Certificate X.509 is send and also server also can send a certificate request. Now there is the Server Key Exchange, where the pre-master secret (*obtained from key exchange, example: using DH is (g^(a*b))*(mod p)) is shared, it will be used later to create the master secret(*obtained from pre-master using a PRF whose output is equal to 48 bytes, from that multiple session keys are derived using PRF, half used for integrity and half for confidentiality, this is based on the principle of don't use the same key for both encryption and certificate signing*).
    - Client Key Exchange, same as server one, and also a certificate is send by the client id server ask for one and an explict message that provides verification of client certificate.
    - Change Cipher Spec message is send to make the transistion of both parties to the agreed cipher suite.

- *Finished* message is sent when handshake is terminated and the entire handshake is hashed. It is used to signal the end of hanshake protocol. After that each message will be encrypted using shared secret.

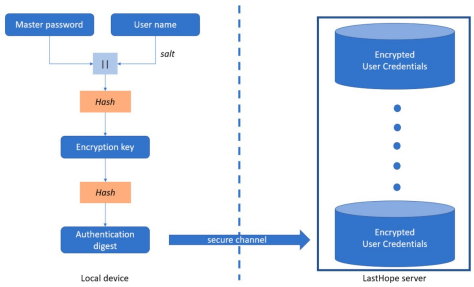
(e)**How TLS ensure integrity, confidentiality and authentication triad**: Confidentiality is ensured using a combination of symmetric and asymmetric encryption. Each secret key is used only for one session, avoiding replay attacks. Moreover TLS (and also SSL) uses asymmetric Key encryption to share the secret key so there is no key distribution problem.
Integrity: it can be ensured only using an hash or MAC algorithm. A MAC algorithm is similar to hash and it's a tag used to ensure that the message came from the stated sender(so also provides authenticity) and has not been modified while in transit. Similar also to digital signatire but a symmetruc is used instead of PKC.
Authentication: server is authenticated by the client using server's public key to encrypt data that are used to compute the secret key, hence server can generate the private key only if he has the correct private key. To authenticate the client, the server uses the client's public key in the client certificate to decrypt data client sends during step 3. Also digital certificates are part of authentication process.

6. In the context of data protection and privacy, (a) give two possible/complementary definitions of privacy and briefly comment on them, (b) explain the linkage attack on anonymized dataset with an example, and (c) explain the trade-off between anonymity and utility in the context of realizing data sets.
   (a)**Privacy**: we can talk about a privacy situation when everyone has control about his personal information. However privacy can be described also as the situation in which there is a high level of difficulty in correlating actions and data.
   (b)**Linkage attack**: it's based on learning sensitive data by joining two datasets on their common attributes. The focus is on *quasi-identifiers,* pieces of information that alone are not sufficient to uniquely identify an entity, but if combinated with others they can uniquely identify an entity. An example is a dataset where an user is recorded “in anonymous way” using the tuple (birthdate, zipcode, gender), in reality there is no possibiliy of anonymity since the tuple above can uniquely identifies more than the 60% of population.
   (c)**Trade-off between anonymity and utility in datasets**: the anonymity-utility trade-off happens because we want both privacy and usability of data and even if we use pseudonyms they must be related in some way to the original identifier, otherwise they are meaningless. So if we focus only on privacy our data become useless, and if we focus only on usability we loss privacy, hence we need to find a balance between the two.

7. Describe a Cross-Site Scripting attack and the main mitigations that can be used to prevent it.
   A Cross-Site Scripting attack, also called XXS attack occurs when an application inlcudes untrusted data in a web page without a proper validation, this can be done using a browser data that can create JavaScript. So XXS also allows attackers to execute scripts on the user browser, in order to redirect him to malicious site or to collect information about the session. Since scripts are run with admin privileges, the attacker can access to very sensitive information. We have 2 possibilities for XXS: reflected *(a constructed url is used)* and stored *(the malicious url is stored in a comment or forum using POST method).*
   A mitigation can be done filtering all parameters coming from GET and POST method, allowing only charcater that don't have a special meaning in HTML and JavaScript. In this case is better to do a positive filtering, listing only admitted characters, rather than not admitted, because it's easy to forget something.

8. LastHope offers a password manager service structured as depicted in the figure. When users create accounts on their devices, they set a *user name* and a *master password*. The former is used as a salt that is concatenated with the latter (see the box labeled with || in the figure) and then hashed to create a digest that is, in turn, used to generate an *encryption key* for encrypting all credentials of a user before sending them to the LastHope server. Then, the *encryption key* is hashed again and an authentication digest is sent to the LastHope server over a secure channel. In the LastHope server, a database of credentials is created for each user where the *authentication digest* is stored together with all the user's credentials. You are

a security expert paid to comment on the password manager service and, in particular, to answer the following questions:
(a)Which of the available hash function algorithms would you advise using in the Local device? Motivate your answer, highlighting advantages and possible disadvantages.
(b)Is it secure enough to use the *user name* as the salt to derive the encryption keys? Motivate your answer and suggest, if the case, possible extensions.
(c)Which security service would you adopt to make the communication channel between the Local device and the LastHope server secure? Motivate your answer and discuss possible configuration problems of the proposed security service.
(d) Which encryption primitive would you suggest adopting by the LastHope server to encipher the credentials in each user database? Motivate your answer.-->password storage algorithms



(a)**Suggested hash algorithm for the local device**:For the local device's hash function algorithm, I would recommend using a strong and secure cryptographic hash function such as SHA-256 or SHA-3. These algorithms are widely accepted and provide a good balance between security and performance. The advantages include resistance to collision attacks and being computationally efficient. However, it's essential to use a hash function that is currently considered secure, as cryptographic algorithms can become vulnerable over time due to advancements in attack techniques.
(b)**Using the username as a salt:**in order to derive encryption keys, is generally not recommended. Salts need to be unique for each user and random, and using the username may not provide sufficient randomness, moreover nobody ensure us about the uniqueness of a username. Instead, it's better to generate a unique random salt for each user during account creation, this can be done using a random generated value with a length between 32 and 64 bytes. This ensures that even if two users have the same password and same username, their encryption keys will be different. We can also rely on this thanks to avalnache effect, which ensure big difference in output even if we have very similar data in input.
(c) **Secure the communication channel between the local device and the server:** I recommend to use a secure communication protocol such as Transport Layer Security (TLS) or its predecessor, Secure Sockets Layer (SSL). TLS provides encryption and authentication, ensuring that the data transmitted between the devices is confidential and has not been modified. Configuration problems may arise if the TLS implementation is outdated, so it's crucial to keep the software up-to-date, moreover we can have some problems due backwards compatibility (*weak cipher suites and broken hash functions are supported for profit reason),* logical flaws and implementation issues related to the possibility to have bugs in the libraries, such as OpenSsl.
(d) **Encrypting the credentials in each user database on the LastHope server** : t's recommended to use strong encryption primitives such as AES (Advanced Encryption Standard). AES is widely accepted as a secure symmetric encryption algorithm since supports variable block and key length, and remain safe even if computer power rising. It provides a good balance between security and performance. The encryption process is based on a series of table lookups and XOR operations, usually 128-bit keys are used.

9. In the context of access control, describe the general architecture of an access control enforcement mechanism including (a) subjects, (b) requests, (c) guard, (d) policy, (e) isolation boundaries, (f) audit log, and (g) the role of authentication and authorization.

(a)**Subject**: is any entity (user, program or device) allowed to ask a resource or a service. An user is a subject identifying with a human being
(b)**Requests**: it can be granted or denied. A subject make it in order to access to some resources and data in a system. All request must be routed to the Guard, in an access control context, this can be done with a good implementation of the outer boundary.
(c)**Guard**: is responsible for grant or deny access evaluating policies. If a subject is authorized then he can perform the actions specified in the request. The by-passing of the guard is possible thanks to the outer boundary.

(d)**Policy**: is a set of rules to implement specific security requests and specifying which actions a subject can perform on resources. Policy are mathematically represented in a formal way in "model", while the low level implementation is done in the "enforcement" part. Policy, model and enforcement are indipendent, so for example reasoning abut policies doesn't require a knowledge about enforcement. This strategy is called separation of concerns, and it's possible thanks to the structured approach to the problem.
(e) **Isolation Boundaries**: they prevent un-authorised access to sensitive pars. We can find two types of isolation boundaries: the outer and the inner. The first one ensure that nobody can by-pass the Guard, which need to evaluate all the requests coming from subjects. Usually only authorized user, like administrator with privileged permissions can by-pass it in order to, for example, update policies or any other administration operations.
The second one is the inner boundary, which protect the Audit Log (explained below); this boundary can never be by-passed, even by admins, since that could lead to integrity attacks.
(f)**Audit Log**: here all past requests with their results are stored and this is protected by the inner boundary who prevent modifications to granted(or denied) authorizations.
(g) **Authentication and authorization** they are two processes that administrators use to protect systems and information. Authentication verifies the identity of a user or service (through different means depending on the assurance level required) and authorization determines their access rights, based on some attributes/privileges that the user has.

10. Define the notions of Confidentiality, Integrity, and Availability.
(a)**Confidentiality**: prevent unauthorised use, disclosure and access of information, permitting only authorized action on them. Moreover data shared in a private channel must remain private to the two or more parties that are communicating, like data in a military context.
(b)**Integrity**: prevent unauthorised modifications of information and permit only authorized ones. This also implies non-repudiation and authenticity, like data of a bank service (we don't want that bill are changed by unauthorised people!)
(c)**Availability:** prevent unauthorised witholding of information or actions that could be limit access to some sensitive information, like ones related to healthcare.

11. The BrokenWare company uses a tool capable of deploying and serving virtual computers. The tool contains a critical flaw that can be exploited to compromise any system. The vendor release a patch for the vulnerability on 23 February 2021, but BrokenWare does not update its systems. At the beginning of February 2023, the company is hit by a ransomware that is able to encrypt all the company data by exploiting the unpatched vulnerability. You are a security expert paid to comment on why it is *virtually* impossible to decrypt BrokenWare's data without paying the ransom.
Specifically, you should explain the reason by first defining the notions..(already done above)
Finally, you should describe (f) which security architecture and best practices could have mitigated this kind of attack [10 points].
Ransonware attacks use cryptography that with two keys to encrypt and decrypt files. The attacker holds the decryption key until you pay the ransom, so it's *virtually* impossible to decrypt crypted data without the decryption key, hence without decryption key.
(f)**Security architecture and best practices to mitigate this ransonware attack**:Implement a ***zero trust architecture*** to prevent unauthorized access to data and services. Adopting a zero trust approach ensures that all access requests are verified and authenticated, this involves verifying the identity of anyone trying to access resources within the network, regardless of whether they are inside or outside the organization. This can be tedoius, but help to minimize the risk of unauthorized access and ***limiting the spread of ransomware.***

12. Explain how to protect a password file with hashing and salting; discuss (a) what is hashing, (b) what is salting, (c) why hashing is not enough for protecting passwords and how salting mitigates the problem of hashing, (d) describe the structure of a password file, and (e) explain if salts can be stored in clear.
(a)**Hashing** is a one-way function that transforms input data (like a password) into a fixed-size string of characters, which appears random. A good hash function should be fast to compute but extremely difficult to reverse. Common hash functions include SHA-256 and bcrypt.
(b) **Salting**: involves adding a unique, random value (32-64 bytes) to each password before hashing. This ensures that even if two users have the same password, their hashed values will be

different due to the unique salt. Salting defends against precomputed attacks, like dictionary attacks, where an attacker has precomputed hashes for common passwords.
(c) **Why Hashing Alone is Not Enough**: If you only hash passwords without salting, identical passwords will produce the same hash value. Attackers can use precomputed tables (dictionary) to look up these hash values and quickly determine the corresponding passwords. This is a vulnerability that salting helps to mitigate.
(d) **Structure of a Password File**: typically contains user-specific information, including the username, hashed password, and the associated salt. The structure might look like:
(e) **Can Salts be Stored in Clear**: Salts should not be stored in clear. The purpose of a salt is to add uniqueness to each password, enhancing security. If salts are stored in clear, an attacker can incorporate them into their attacks, defeating the purpose of salting. It's essential to keep salts confidential and unique for each user.

13. In the context of the SAML standard, (a) explain its main goal, (b) describe the two scenarios for the Web Single Sign On profile, namely Identity provider initiated and Service provider initiated, (c) explain how is it possible to guarantee trust in the assertions consumed by the relying party (hint: remember that assertions are cryptographically signed), and (d) explain how it is possible to avoid man-in-the-middle attacks.
(a)**Main goal of SAML**: it stands for Security Assertion Markup Language, it is a markup language used to exchange authentication and authorization metadata between parties, usually between IdP and SP. One of the most important use case is in web-browser single sign-on, in order to create a security domain trusting a unique and common security token generated by the IdP.
(b)**Two scenarios for the Web Single Sign On**: when we use SAML for single sign-on we can have two different scenarios, depending on who starts the process:
    - IdP starts the process: the user login in a trusted website and then is redirected over to a partner web site (example of airline company and car rental company). The IdP asserts that the user is known and has authenticated correctly, andh has the privileges to access to the partner site
    - SP starts the process: this is the most common one. The user wants to access to a service on a SP, but since these require specific authorization or authentication, the SP redirect the user to the IdP with an authentication request in order to have the user logged in. after that, the IdP produces an assertion informing the SP that the user has correctly authenticated but is a subject of the SP to check if the user has the authorization to access to the required resource.

    (c)**How guarantee trust in the assertions consumed by the relying party**: to provide trust between the two parties (SP and IdP), the use of PKI is strongly recommended, to provide digital signatures:Assertions are signed by the issuing party (usually the identity provider) using a private key. The relying party (service provider) can verify the signature using the identity provider's public key, ensuring the integrity and authenticity of the assertion. Generally speaking also the following should be granted in every message: expiration date, unnique id and messages should only be accepted by the SP application they are intended for.
    (d)**How avoid man-in-the-middle attacks**: TLS and digital certificates

14. Explain the notion of pseudo anonymization function, describe at least two possible **implementations** together with their advantages and disadvantages.
    A pseudo anonymization function is a function that provide the possibility to maps identifiers to pseudonyms, it is required that given two different input even their pseudonyms are different, however a single identifiers can have multilple pseudonyms as long as the reverse process is possible. The reversion can be done using the pseudonymization secret. We can implement this kind of function in 4 different ways:
    - use of counter: identifiers are substituted by a monotonic counter. Pros: simple to implement, cons: requires lot of space and the fact of being sequential leads to get information about order of the data.
    - Pseudo random generator, similar to previous but avoid correlation due to sequential order, but it's difficult to implement since we need to avoid repetitions and not resolve scalability problem.
    - Using cryptographic hash functions, the pseudonyms is the digest of the identifier, not difficult to compute but it suffers of dictionary and brute force attacks.

- Encryption, using block cipher like AES, the identifier is encrypt using the secret key, which, since AES is a symm. technique, the key is both the secret used for pseudonymization and recovery.

15. In the context of access control, explain (a) explain what is a confused deputy with the help of an example, (b) how capabilities allow for avoiding a confused deputy attack, (c) explain the notion of Discretionary Access Control (DAC), (d) explain the notion of Mandatory Access Control (MAC), and (e) discuss the main advantages and disadvantages between DAC and MAC.
(a)**Confused deputy**:it is a security concept that refers to a situation where a trusted entity (deputy) is manipulated into performing actions on behalf of an attacker without the deputy's awareness. This manipulation can lead to unintended and potentially malicious consequences, as the deputy may inadvertently carry out actions that it has the authority to perform, but are performed on the behalf of the attacker. Example of a compiler service: a subject that can execute the compiler service could gain access to resources without having permissions. The attacker can delegate the compiler to read its ource file "*In"* , at the same time the compiler is allowed by the OS to write the file "Bill" where a billing is contained. In this way the attacker can indirectly write the file "Bill" exploited the fact that the compiler doesn't know from which source each rights derives.
(b)**How capabilities allow for avoiding a confused deputy attack**: they avoid the confused deputy problem because now permissions are associated with the subjects and not with the resources, so now compiler can distiguish the sources from which rights derive, however capabilities are more difficult to implement, but they are more secure.
(c)**Discretionary Access Control (DAC)**: subjects give rules to other subjects, so for example the owner of a resource decides how this can be shared. Usually users are grouped in order to minimize space used by the AC matrix or AC list. When an user must be authorized, his category must be check and verified that has the proper rights.
(d)**Mandatory Access Control (MAC)**: more restrictive than DAC, MAC is an access control model where access decisions are based on security labels assigned to both subjects and objects. The access control decisions are made by a central authority, and users have less discretion in determining access permissions.
(e)**Advantages and disadvantages between DAC and MAC**:
    - DAC: pro: flexible and easy to realize. Cons: subjective to owner decisions to modify access rights, can be inconsistent due to user errors, vulnerable to trojans: maliciuos program run by authorized users but on the behalf of an attacker, in order to obtain information or even modify them.
    - MAC: pro:centralized control over resources guarantees a higher level of control and a more secure and robust system. Cons: less flexible, as access decisions are based on predetermined policies. Implementing and managing MAC systems can be more complex, especially in dynamic environments.

16. Describe the technique of k-anonymity and explain how it can mitigate linkage attacks.
K-anonymity allows that information for each person in a database cannot be distinguished from at least k-1 individuals, whose information also appears in the database. So each record must be indistinguishable from at least k-1 other records, since each quasi-identifiers(identiers that not completely identifies a person, but can do it if used together with other other quasi-identifiers) can appear in the table only k times, the most high is k the most indistinguishable identifiers are. We can obtain k-anonymity with generalization (which consists in replace quasi-id with less specific ones but still semantically consistent values until we obtain k identical element: example->age=20,21,22,23.. are generalized writing 2*) or suppression (totally remove an attribute, also used when generalization causes too much data loss so it's meaningless keeping information about that attribute on the database). However k-anonymity suffers of the privacy-usability trade-off.
Thanks to k-anonymity we can mitigate linkage attack-->learning sensitive data from joining two database. In a linkage attack the attaccker exploit the potential of quasi-identifiers: if they are used together and combined they can identify even uniquely an user. For example the tuple (birthdate, zip code and gender) can uniquely identify about 2/3 of american population, so we can easily imagine how in smaller context this could be even more efficient and precise.

17. Describe an SQL injection attack. Which are the main mitigations to an SQL injection attack?

An SQL injection attack concerns obtaining information without cracking any password. It is based on the fact that untrusted data are sent to an interpreter as a part of a command or a of query, so the interpreter is tricked by the attcaker to execute the maliciuos command allowing the attacker to access data without a real authorization. A common example is based on the fact that the query string **username='admin';- -** is interpretated as **select * from USER where username='admin'; -- password='admin-pass'**. So only adding the comment command - - the attcker can by-pass the password control since every things written after the commen is ignored. This leads to a risk in confidential but also in integruty since data as they can be read can also be modified in similar way. A good mitigation for this problem is based on the fact that usually commands are executed with root priviliges, hence we need too limit actions that can be performed by our application. This can be done applying the **principle of least privilege**: giving to an app, a user or a process only those privileges which are essentially vital to perform its intended functions, and nothing more. Moreover an **input sanitations** should be done, allowing only parametrized and pre-compiled queries.

18. Define the notion of hash function and its main properties: (a) Ease of computation, (b) Compression, (c) One-way, (d) Weak collision resistance, and (e) Strong collision resistance.
(a) **Ease of computation**: easy to compute, but difficult to reverse...
(b) **Compression**:h maps inputs x of arbitrary bit-length into an output h(x) of a fixed bit-length n.
(c) **One-way**:given y is computationally infeasible to find an input x such that h(x)=y.
(d) **Weak collision resistance**: given an input x and h(x) is computationally infeasible to find x', different from x, such that h(x)=h(x').
(e)**Strong collision resistance**: is computationally infeasible to find any two different x and x', such that h(x)=h(x'). The difference between the weak collision resistenc is that we don't have any initial values.

19. Explain the notion of Multi-level security by illustrating it with the Bell-La Padula model; in particular discuss the notions of (a) security level, (b) need-to-know, (c) dominance relation, (d) no-read-up rule, (e) no-write-down rule.
(a)**security level(S)**: is defined by an orderd set, such as Top secret>secret> confidential>unclassified. They define what grade of security should be granted to a specific resource.
(b)**need-to-know(N)**: the need-to-know categories are a unordered set expressing some different categories that can have different interest in the resources, for example we can have *Nuclear, Crypto, Personnel,..* categories. This ensure also the principle of least privilege since only authorized categories can access to specific topic. (a) and (b) together form L=(S,N), security labels
(c)**dominance relation**: in order to define which user is authorized to access to resources, each user is associated with a clearance (also called authorization level), C=(S,N), where S I sthe **hierarchcal security level** inidcating the degree of trustworthiness the user has. So a dominace relation can be constructed between user and resources, basing on sensitivity labels of the two. This is directly related to the rules below. For example L1=(S1,N1) dominates L2=(S2, N2) only if S1>=S2 and N2 is a subset of N1.
(d)**no-read-up rule**: a subject can read a resource R only if Ls dominates Lr, the clearance of th euser must dominate than sensitivity label of the resource. Confidentiality is guaranteed.
(e)**no-write-down rule**: a subject can write a resource R only if Lr dominates Ls, so the clearance of the user is dominated by the security label of the resource. This is ensure confidentiality preventing unauthorized disclosure of info to lower level by an higher level user. Totally doesn't care about integrity, since low clerance user can write higher level resources. To resolve the lack of integrity we can combine the Bell-LaPadula (based on the two rule above and tranquillity principle--> security labels shouldn't change freqeuntly) with the Biba model: it's based on integrity labels: the one of the resource characterizes the degree of trustworthiness of the resource, while the one of the subject measures the confidence that one places in its ability to produce or handle info.

20. Briefly describe the notion of Data Protection Impact Assessment (DPIA) according to the Art. 35 of the GDPR.
The art. 35 of the General Data Protection Regulation states that when a type of processing of data using the technology is likely to result in a high risk to the rights and freedom of a person, then the

controller, before starting thhe process, shall produce an assessment of the impact that the upcoming process could have on personal data.

21. Define the notion of cryptosystem.
    A cryptosystem is a 5-tuple system, cryptosystem=D(E(M,K),K), where E and D are respectively the encryption and decryption algorithms, M is the plaintext and K is the encryption key. The K used in encryption and decryption could be different (asymmetric cryptography) or not (symmetric cryptography)

22. Explain the framework of Attribute Based Access Control (ABAC), in particular how an access request is processed, and the reasons for which it has been introduced with respect to previous models (such as RBAC).
    ABAC was introduced because RBAC wasn't enough for expressing authorization depneding not only on roles but also on additional attributes of subjects, environments and resources. With ABAC we don't totally lost roles but they become just a specific category of attributes. We have a more flexible and dynamic system since policies can be writtem according a large number of attributes, and more complex relationship between subjects, resources and environment can be done.
    In the ABAC framework authorizations depends on 3 different types o attributes: user or subjects attributes, resource attributes and environment attributes (like hour, day of the week..).
    When an access request is made, the ABAC system evaluates the relevant attributes associated with the subject, resource, and environment, based on the defined policies.
    The system checks if the attribute values satisfy the conditions specified in the policies, if the conditions are met, access is granted; otherwise, it is denied.
    - Policy decision point receives the request, previously converted in xacml from its native format by the context handler, so it can do proper comparisons with the policies that can be applicated on it, and returns an authorization decision based on policies . A response object is created and send to context handler who converts it in its native format from xacml, this object can also include obligations (actions that must be carriedd out before or after an access request is granted or denied).
    - The policy enforcemnet point enforces the decision and it can honor all obligations or treat the decision as denied.