Department of Computer Science

Computer Security

Exam 1

February 19, 2024

Solutions & Discussion

Prepared with tender loving care by Paul Krzyzanowski

100 POINTS – 25 QUESTIONS – 4 POINTS EACH – For each statement, select the *most* appropriate answer.

1. A program with a large *attack surface* will:
   (a) Contain a large number of vulnerabilities.
   (b) Provide the adversary with many different ways to try to attack the system.
   (c) Be less likely to be impacted by a denial of service (DoS) attack.
   (d) Be at risk of attacks by malicious insiders.

   An attack surface refers to the sum of possible attack vectors in a system. It is the total number of places in a system that an attacker might use to try to get into the system. It does not mean that these are vulnerabilities in the system.

2. A specific problem with attacks by malicious insiders is:
   (a) Some of the company's security policies don't guard against them.
   (b) They are more skilled than outsiders.
   (c) They can use deception.
   (d) They can use techniques of usurpation.

   Not (b): People outside the organization are often more skilled than people within the organization.

   Not (c) or (d): People inside the organization *might* be able to use deception more easily (e.g., pretending to be a system administrator or manager) or usurpation (taking unauthorized control of a system, such as theft of data or misuse of system privileges). Both of these may be easier for insiders but are not specific problems to insiders.

   (a) A *specific* problem to malicious insiders is that security policies (physical access and access to resources) may not be a barrier.

3. What best describes the role of the Trusted Computing Base (TCB) in a secure computing environment?
   (a) To ensure that a system's security policies operate correctly.
   (b) To provide a backup solution for data in case of a cybersecurity breach.
   (c) To manage user identities and access permissions within an organization.
   (d) To encrypt data transmissions over unsecured networks.

   The Trusted Computing Base refers to all the hardware and software of a computing system that is essential to its security. It means we trust the firmware, bootloader, operating system, device drivers, databases, etc. – and their ability to enforce security policies. If you don't trust the TCB, then you cannot trust the security of the environment.

4. What does *Schneier's Law* state about the design of cryptographic systems?
   (a) The strength of a cryptographic system lies in the complexity of its algorithm.
   (b) Encryption keys should be large and random.
   (c) Any person can invent an encryption algorithm that they themselves cannot break.
   (d) All ciphers eventually get cracked, which is why it's important to update systems to use the latest algorithms.

   Schneier's Law (not a real law) states that any person can invent a cryptography algorithm so clever that she or he can't think of how to break it. That does *not* mean it is a secure algorithm. It's a lesson to not create your own security algorithms and protocols.

5. Which of these is ***not*** a desirable property of a good cryptographic algorithm?
   (a) The algorithm should be a well-protected secret.
   (b) There should be no way to extract the plaintext from the ciphertext except through an exhaustive search.
   (c) The ciphertext should have high entropy.
   (d) The size of the resulting ciphertext should be approximately the same as the plaintext.

   A key principle in cryptography is Kerckhoffs' Principle. It states that *a cryptosystem should be secure even if everything about the system, except the key, is public knowledge.*
   Items (b), (c), (d) are all desirable properties of a cryptographic algorithm.

6. If it takes a year to test all keys but you found a relationship between the ciphertext and two bits of the key, approximately how long will it take you to test all keys?
   (a) 1 month.
   (b) 3 months.
   (c) 4 months.
   (d) 6 months.

An example of finding a relationship between the ciphertext and two bits of the key could be something like "if the first and last bytes of the ciphertext are 0, then bits 4 and 12 of the key must be 1." This will never be the case with a good cipher but, if a condition like this exists, then we have fewer keys to search in a brute-force attack.

Specifically, by being able to predict the values of two bits of the key, we have two fewer bits to test, which means a factor of 4 (2^2). Thus, we have $1/2^2=1/4$ the number of keys to try. Instead of taking 12 months to test all keys s, it will take 12/4=3 months to test ¼ the number of keys.

7. Why does it make more sense to compress a file before encrypting it instead of after encrypting it?
   (a) It prevents an attacker from uncompressing the contents.
   (b) Encryption hides the fact that the file is compressed.
   (c) Compression is mostly ineffective on encrypted files.
   (d) Most compression algorithms are optimized for encrypted data, resulting in higher compression ratios.

Data compression works by removing redundancy. Lossless compression changes data to create artificial redundancy that can then be removed. A good cipher produces output that appears completely random; there are no patterns in the data and no redundancy. Hence, compression will be ineffective: the "compressed" file will be approximately the same size as the orginal.

Not (a) If the contents are encrypted, they be secure whether compressed or not.
Not (b) Why should it be a secret that the file is compressed?
Not (d) This makes no sense!

8. A frequency analysis reveals that the ciphertext contains the same distribution of byte values as the plaintext. The cipher used was most likely:
   (a) A monoalphabetic substitution cipher.
   (b) A polyalphabetic substitution cipher.
   (c) A columnar transposition cipher.
   (d) A rotor machine.

Not (a): A monoalphabetic substitution cipher will produce the same distribution of frequencies – but for different byte values.

Not (b) A polyalphabetic substitution cipher will destroy the frequency distribution since one byte of plaintext will result in different bytes of ciphertext based on its position.

Not (d) A rotor machine implements a polyalphabetic substitution cipher.

A transposition cipher does not change the values of the bytes – the frequency that each byte value occurs will be the same. The bytes are just scrambled in a different order.

9. The *one-time pad* is rarely used because:
   (a) Managing and distributing keys is difficult.
   (b) Encryption was relatively slow and faster ciphers have been developed.
   (c) Modern ciphers are at least as, if not more, secure.
   (d) It does not provide diffusion.

The one-time pad provides perfect secrecy. However, this is achieved only if the key is truly random and at least as long as the message. This makes the one-time pad impractical because:
1. Key Distribution Problem: The OTP requires that both the sender and receiver have access to the same random key, which must be as long as the message itself. Securely distributing and managing these keys, especially for large volumes of data or over long distances, is impractical.

2. Key Management: Each key can only be used once to maintain absolute security. This necessitates a large supply of keys for ongoing communication, complicating key management and storage.

3. Lack of Practicality: For most applications, the logistical challenges of generating, distributing, and securely storing large quantities of random keys outweigh the benefits of the unbreakable encryption that the OTP offers.

10. What is the primary weakness of the Electronic Codebook (ECB) mode of operation in block ciphers?
    (a) It requires an initialization vector (IV) for every encryption operation, complicating the encryption process.
    (b) It encrypts identical plaintext blocks into identical ciphertext blocks, making it vulnerable to pattern analysis.
    (c) It is significantly slower than other modes of operation due to the extra encryption required for each block.
    (d) It cannot support the concurrent encryption of multiple blocks of data.

In ECB mode, identical plaintext blocks are encrypted into identical ciphertext blocks with the same key. This characteristic makes ECB vulnerable to pattern analysis and could potentially reveal information about the plaintext data, especially with data that has repetitive patterns or structures, such as images or documents with recurring headers. This makes ECB mode unsuitable for encrypting large amounts of data or data with predictable or repetitive patterns, as it does not sufficiently disguise the underlying information.

Not (a): It does not use initialization vectors.
Not (c): There is no extra encryption or any extra work. It's the fastest of the cipher modes.
Not (d): Each block can be encrypted independently with no need for the results of other blocks.

11. What is the purpose of performing multiple rounds in an SP network within block ciphers?
    (a) To create symmetry and ensure that the encryption process can be easily reversed during decryption.
    (b) To enable a smaller key size by reusing the same key in different forms across multiple rounds.
    (c) To maximize efficiency by minimizing the computation in each round.
    (d) To spread the influence of each plaintext bit over the entirety of the ciphertext block.

The purpose of multiple rounds in a substitution-permutation network is to increase the amount of confusion and diffusion in each block:
Diffusion: Uses permutations spreads out the bits of the plaintext over the entire ciphertext, ensuring that a change in a single bit of the plaintext results in multiple changes in the ciphertext. This makes it more difficult for an attacker to deduce specific plaintext bits based on the ciphertext.
Confusion: Uses substitutions to replace bits in a way that the relationship between the ciphertext and the key is as complex as possible. By doing so, it obscures the relationship between the key and the ciphertext, making it harder for attackers to perform analysis that might reveal the key or plaintext.

Each round enhances the amount of confusion and diffusion.

12. An advantage of Elliptic Curve Cryptography (ECC) over RSA is:
    (a) ECC is suitable for both encryption and digital signatures.
    (b) ECC is resistant to quantum computing attacks.
    (c) ECC does not rely on trapdoor functions.
    (d) ECC is faster and uses shorter keys for a comparable level of security.

Not (a): RSA encryption serves the same purpose.

Not (b): Both ECC and RSA are vulnerable to future quantum computation attacks.

Not (c): All public key algorithms rely on trapdoor functions: being able to decrypt (apply an inverse) if you know extra information.

ECC is faster (particularly for key generation and decryption) than RSA. Point multiplication is more efficient than modular exponentiation.
ECC can use also much shorter keys for the same level of security.

13. For Alice to create a message that only Bob can read, she would encrypt it with:
    (a) Alice's private key.
    (b) Alice's public key.
    (c) Bob's private key.
    (d) Bob's public key.

Whatever is encrypted with a public key can only be decrypted with the corresponding private key. Whatever is encrypted with a private key can only be decrypted with the corresponding public key.

Bob is the only one who has Bob's private key. Hence, anyone can encrypt a message with Bob's public key and only Bob will be able to decrypt it since nobody else will have Bob's private key.

14. What fundamental problem does the Diffie-Hellman algorithm solve in cryptography?
    (a) It provides a method for digital signature verification.
    (b) It encrypts messages with a symmetric key algorithm.
    (c) It creates a public and private key pair for use in asymmetric encryption.
    (d) It offers a secure way to create a shared key over an insecure channel without prior secret sharing.

The Diffie-Hellman algorithm allows two parties to agree on a common key without exchanging secret information.

15. *Forward secrecy* is a property that:
    (a) Makes it impossible to recover plaintext without knowledge of the key.
    (b) Protects past communications even if an attacker compromises the users' long-term keys.
    (c) XORs each message with the ciphertext of the previous message before encrypting it.
    (d) Requires a key to be at least as long as the message that is being encrypted.

Forward secrecy means that even if someone gets the encryption keys later, they can't decrypt past messages.

A long-term key is a key that you use over and over again, such as one tied to your identity.

By using that only for authentication and then an algorithm like Diffie-Hellman with randomly-generated keys (ephemeral keys), it will be impossible for an attacker to derive keys used in other communication sessions.

16. A *hybrid cryptosystem*:
    (a) Uses a public-key cryptosystem to send a symmetric key.
    (b) Employs multiple rounds of encryption for stronger security.
    (c) Uses two or more layers of encryption algorithms on a message to guard against one of them getting attacked.
    (d) Combines ciphertext with a message authentication code (MAC).

A hybrid cryptosystem is any cryptographic system that combines symmetric and asymmetric encryption methods.
It is typically used to use a public key algorithm (such as ECC or Diffie-Hellman) to send or create a session key that will then be used to encrypt data in a symmetric algorithm.

17. How does CBC-MAC ensure message integrity?
    (a) By encrypting the message in CBC mode and outputting only the final block as the MAC.
    (b) By encrypting a hash with the receiver's public key, so only the recipient can authenticate the message.
    (c) Through hashing the message content and encrypting the hash value with a block cipher in CBC mode.
    (d) By applying a hash function to a message encrypted using a symmetric block cipher in CBC mode.

Instead of relying on a hash function, like HMAC does (hashing a message and the key), CBC-MAC generates a fixed-size message authentication code by using a block cipher in CBC mode.

In CBC mode, each block is XORed with the previous ciphertext block. In CBC-MAC, every block is discarded and the final block is used as the message authentication code.

18. A MAC differs from a digital signature because a MAC:
    (a) Uses public key cryptography.
    (b) Allows you to reconstruct the message.
    (c) Authenticates the creator of the message.
    (d) Can be validated by only a limited group of trusted principals.

Message Authentication Codes (MACs) use shared keys. Digital signatures use public key cryptography. Anyone with the shared key can create or validate a MAC. Only the owner of the private key can create a digital signature, which can be validated by anyone with the corresponding public key (which is generally not secret).

19. If an attacker steals your X.509 digital certificate, they will be able to:
    (a) Encrypt messages that only you can decrypt.
    (b) Decrypt old encrypted messages that you created.
    (c) Sign messages masquerading as you.
    (d) All the above

There is no need to "steal" a digital certificate – they do not contain secret data.

If someone gets your certificate, they simply have your identity - public key mapping. In other words, they know whose public key they have. All they can do with your public key is:

- Decrypt anything you encrypted with your private key (e.g., validate anything you signed with your private key)
- Encrypt content that can only be decrypted with the corresponding private key.

(b) looks like a plausible answer since an attacker would be able to decrypt messages that you encrypted with your private key. However, the underlying assumption with public-key cryptography is that public keys are not secret, and it makes no sense for you to encrypt content with your private key since there will be no confidentiality as anybody will be able to decrypt that content. This is why encryption with a private key is used for encrypting hashes to create digital signatures – so that anyone can decrypt the signature and validate the hash, knowing that only the owner of the public key could have created it.

20. How does the use of a trusted third party solve the key explosion problem?
    (a) Only the trusted third party needs to store the keys for every group that may need to communicate.
    (b) It uses public key cryptography to send encrypted session keys.
    (c) Aside from using temporary session keys, each user only needs to store their key.
    (d) All communications flow through the trusted third party, which acts as a proxy.

A trusted third party reduces the key explosion problem by securely sharing a unique key with each pair of communicating parties, instead of each party needing keys for every other party.

21. One purpose of adding a *nonce* to some key exchange protocols is to:
    (a) Identify the sender of the message.
    (b) Be able to check that a set of messages are related.
    (c) Improve security by increasing the amount of ciphertext that's created.
    (d) Guard against man-in-the-middle attacks.

A nonce is a random bunch of bits. Nonces are primarily used in two ways in cryptographic protocols:

(1) Prevent replay attacks by ensuring that a response corresponds to a request because it contains the same encrypted nonce. We saw this in the Needham-Schroeder algorithm.

(2) Authenticate by having the other side prove that they can encrypt a nonce or decrypt an encrypted nonce.

It does not guard against man-in-the-middle (MITM) attacks because a nonce, by itself, does not establish the authenticity of the communicating parties.

While a nonce helps prevent replay attacks by ensuring that each message is unique, it does not verify the identity of the sender or protect against an attacker intercepting messages and impersonating one of the parties involved. To guard against MITM attacks, additional mechanisms such as digital signatures or mutual authentication are necessary to ensure the authenticity and integrity of the communication.

22. Alice gets a ticket from Kerberos to talk to Bob. She can think of this *ticket* as:
    (a) A message that contains Bob's secret key and is encrypted for her.
    (b) A digital certificate that contains her ID and her secret key.
    (c) A message that uniquely identifies Bob on the network.
    (d) A session key for communicating with Bob that she cannot decrypt.

A Kerberos ticket is data that Alice cannot decrypt. It is encrypted for Bob and contains a session key for Alice and Bob to use for secure communication. Alice received the same key from Kerberos encrypted with her secret.

23. What do *salted hashes* prevent an attacker from doing that she would be able to do with normal hashed passwords?
    (a) Perform a dictionary attack to find a password.
    (b) Do a brute-force attack to find a password.
    (c) See if two or more users share the same password.
    (d) Decrypt the hashed password to reveal a password.

Salt prepends a random string to a password prior to hashing it. Salt is not secret – it is stored with the password hash.

Salt makes precomputed hashes impractical. You cannot store a table of hash values of all likely passwords combined with all possible random strings.

Instead, an attacker will be forced to do (a) a dictionary attack on individual hashed passwords, testing hash(salt+password) for various password in the dictionary. Optionally, an attacker may choose to do (b) a brute-force attack, testing every possible password prepended with the salt.

Because the salt strings are random for each password, two identical passwords from two different users will have different salt values (they're random), resulting in different password hashes

24. Which of the following is *NOT* an example of multi-factor authentication?
    (a) A fingerprint and a PIN.
    (b) A password and an answer to a security question.
    (c) An access card and a PIN.
    (d) A fingerprint and a one-time password.

Factors are (1) something you have, (2) something you know, (3) something you are. Multifactor authentication *must* combine two or more of these factors. A password and an answer to a security question are both the same factor: something you know.

25. Which authentication technique does not rely on a client sharing a secret?
    (a) Challenge Handshake Authentication Protocol (CHAP).
    (b) Time-based One-Time Passwords (TOTP).
    (c) Kerberos.
    (d) Passkeys.

In CHAP authentication:

Bob sends Alice a nonce, *n*.

Alice returns hash(n, *key*)

Bob also computes hash(n, key) and compares it with Alice's value.

TOTP authentication:

Alice sends hash(key, time)

Bob computes hash(key, time) and compares it with Alice's value.

Kerberos = trusted third party

Kerberos sends Alice a session key encrypted with her secret key.

To do this, Kerberos has everyone's secret keys.

Passkeys: use public key authentication:

Bob sends Alice a challenge (a nonce, *n*)

Alice sends signature(*n*), which is effectively the hash(*n*) encrypted with her private key.

Bob validates the signature since he has her public key stored. Public keys are not secret and he obtained Alice's public key during the passkey setup process.

The end.