Home assignment

Here is a comprehensive solution. If your focus is solely on the high-level overview, you may choose to skip steps 1 to 4.

Q1 - Step 1: Selecting the Proxy Software

Low-Level:

- **Choose Proxy Software:**

  On-Premises Proxy Servers: Deploying proxy servers in your infrastructure to inspect HTTPS traffic.

  Pros: Complete control, high customization, potentially lower latency.
  Cons: Complex setup and maintenance, scalability challenges, higher upfront costs.

  Cloud-Based Secure Web Gateways (SWG): Using cloud services to enforce security policies and inspect traffic.

  Pros: Easy to deploy and maintain, scalable, good for distributed teams.
  Cons: Less control over data, dependent on internet connectivity, ongoing costs.

Step 2: Network Configuration

Low to Mid-Level:

- **Redirect Traffic and Ensure Transparency:**

Strategic Configuration

PC <----------> Switch <----------> Router (Gateway) <----------> My Proxy <-> WWW

pros\cos: Control for one device. Limited Visibility Covers all

This setup ensures that all internet-bound traffic is transparently monitored and controlled by the proxy, without requiring any specific configurations on the individual PCs. It's a seamless approach to manage web traffic efficiently and securely.

Step 3: Step 3: SSL/TLS Interception Setup

Mid- Level:

- **Generate a Root CA Certificate:** Set up the proxy server to create its own Certificate Authority (CA) certificate. This certificate is key to issuing secure, proxy-generated certificates for HTTPS sites accessed through the proxy, enabling it to decrypt and inspect web traffic.
- **Install the CA Certificate on Client Devices:** Distribute and install the proxy's root CA certificate across all employee devices. This crucial step ensures that these devices recognize and trust the proxy-issued certificates, preventing security warnings in web browsers.

- Certificate holds a user's public key and confirms their claimed identity with a digital signature.

- The CA doesn't validate the accuracy of the user's information, only their identity.

Step 4: Configure the Proxy for SSL/TLS Interception and Content Filtering

Mid to High - Level:

- **Enable SSL/TLS Interception:** Configure the proxy to decrypt and then re-encrypt HTTPS traffic for inspection.
- **Content Inspection and Filtering:**
    - **Detect Keywords:** Search for specific words like "Clown" in the decrypted traffic.
    - **Take Action:** If a keyword is found, block the content or redirect it as per your rules.
    - **Re-encrypt and Forward:** Secure the content again with a new certificate and send it to the user.

Step 5: Deployment and Testing

High-Level:

- **Deploy the Proxy:** Set up the proxy server within your network infrastructure, ensuring it's hosted on a robust server capable of handling the entire company's web traffic efficiently.
- **Testing:** Conduct comprehensive testing to validate the proxy's functionality:
    - **Functionality Check:** Confirm that the proxy can effectively intercept, decrypt, and inspect traffic from commonly accessed websites (like Google or Facebook) and accurately filter out specific content (e.g., instances of the word "Clown").
    - **Load Testing:** Determine the maximum traffic load the proxy can handle without performance degradation.
    - **Security Tests:** Include SSL stripping resistance (to ensure HTTPS traffic isn't downgraded to HTTP), resilience against MITM attacks, and thorough certificate validation checks, focusing on handling of expired or revoked certificates.
    - **Compatibility Testing:** Ensure seamless operation across various browsers (Chrome, Safari, Firefox) and devices (VPNs, smartphones, etc.). Also, test the proxy's functionality in scenarios like simulated server failures to assess its reliability (backup proxy).
- **Monitor and Adjust:** Continuously monitor the proxy’s performance. Be proactive in tweaking filtering rules based on real-world traffic and user feedback. Address any connectivity issues or anomalies reported by users to fine-tune the proxy settings for optimal balance between security and accessibility.

<u>Certificate Pinning</u>

Objective of Certificate Pinning: Certificate Pinning plays a crucial role in bolstering security for applications and web browsers. Its primary aim is to thwart Man-In-The-Middle (MITM) attacks, where unauthorized entities attempt to intercept communications between a user and a website.

Mechanism of Operation:

- **Pre-Stored Certificates**: Certificate Pinning involves pre-storing the trusted certificate of a website within the application or browser. (Think of this certificate as a digital ID card that the website presents to prove its identity.)

- **Verification Process**: Upon attempting to access the website, the application verifies the presented certificate against the pre-stored one. A match confirms the legitimacy of the website, ensuring a secure connection. Conversely, a discrepancy raises a red flag, prompting the application to block the connection to prevent potential MITM attacks.

Impact on Monitoring Solutions:

Challenges for Content Monitoring Systems:

- **Operational Methodology of Monitoring Systems**: Content monitoring solutions scrutinize encrypted traffic by positioning themselves as intermediaries. They decrypt the traffic to inspect its contents (e.g., searching for specific keywords like "Clown") before re-encrypting it with their certificates.
- **Conflict Arising from Certificate Pinning**: Certificate Pinning expects the original certificate of the website. However, the monitoring system's intervention alters the expected certificate, presenting a different one to the application.
- **Consequential Security Concerns**: This certificate mismatch, adhering to the principles of Certificate Pinning, signals a potential security compromise to the application, akin to a MITM attack scenario. Consequently, the application may refuse to establish the connection.

Solutions to Overcome Monitoring Conflicts:

Strategies for Resolving Certificate Pinning Conflicts:

1. **Application Whitelisting and Blacklisting**: Implement a strategy to whitelist applications that do not require certificate pinning checks and blacklist those that are critical for security. This approach allows certain applications to bypass pinning requirements when being monitored.
2. **Certificate Trust Adaptation for Monitoring:**

   Adjusting App Trust Settings: Change your apps to trust the monitoring system's certificate. This means adding it to the app's trusted list, allowing the app to securely connect even when monitored.

   Overcoming Pinning Issues: When your app connects and sees the monitoring system's certificate instead of the expected one, it will still trust the connection, avoiding the usual security blocks from certificate pinning.

   Smooth Communication: This trust ensures your apps can communicate safely, even during inspection, keeping security tight without disrupting monitoring.

   Key Considerations:

   - **Security Management**: Keep the trusted certificates list secure to avoid breaches.
   - **Certificate Upkeep**: Regularly update and check the monitoring certificates for integrity.

**I delved into researching and analyzing proxy servers (of/de-fensive)

Q2 - John and Joan

John can send a **secure** love letter to Joan while ensuring she can **verify** him as the sender.

PGP (Pretty Good Privacy): Proprietary, widely used in business, can be costly, closed source.

Pros: Has a long-standing reputation for security.

Cons: Licensing costs, Closed source, so its code isn't available for public audit.

GPG (GNU Privacy Guard): Free, open-source, compatible with PGP, less common in corporate environments.

Pros: Open source and free to use, Compatible with PGP, Regularly updated and improved by the community.

Cons: Less commonly used in corporate settings. might lag behind in terms of the availability and sophistication of additional tools and integrations developed for it.

Pre-requisites:

1. **Encryption Software**: Both John and Joan need to download and install encryption software like GPG.
2. **Key Generation**: They both generate a pair of keys:
    - **Public Key**: This can be shared with anyone and is used to encrypt messages to the key owner or verify their digital signature.
    - **Private Key**: This is kept secret and is used to decrypt messages received or to sign messages sent.
3. **Exchange of Public Keys**: John and Joan can exchange their public keys in a non-secure way, such as posting them on Facebook (One-Way Function). The security of messages encrypted with public keys depends on the secrecy and security of the private keys, not the public ones.

The Process:

1. **Writing the Letter**: John writes his love letter to Joan on his computer.
2. **Encrypting the Letter**: Using his encryption software, John encrypts the letter using Joan's public key. Now, only Joan can decrypt this letter with her private key.
3. **Signing the Letter**: John then uses his private key to digitally sign the encrypted letter. This signature will allow Joan to verify that it's John who sent the letter.
4. **Sending the Letter**: John sends the encrypted and signed letter to Joan via email or any other digital means.
5. **Joan Receives and Reads the Letter**:
    - **Verifying the Sender**: Joan first uses John's public key to verify the digital signature. If it checks out, she can be confident that the letter is indeed from John.
    - **Decrypting the Letter**: Joan then uses her private key to decrypt the letter and read its contents.

before delving into question 3, it's advisable to explore question 4 first. Grasping the concept of NTLM Relay offers valuable insights into the evolution of Windows authentication mechanisms.

Understanding Kerberos Authentication and Mitigating Pass-the-Ticket Attacks

Introduction

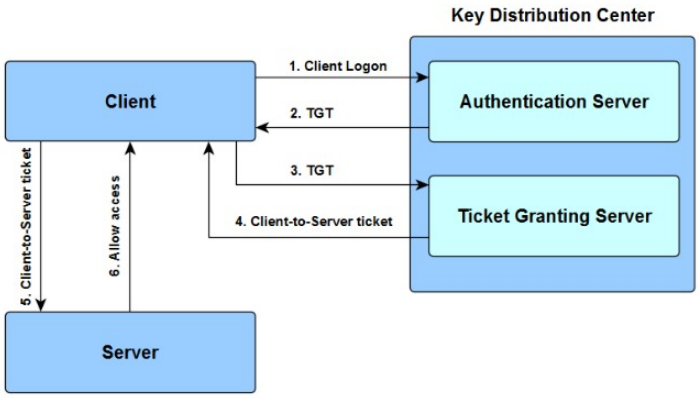
Kerberos is a sophisticated network authentication protocol that uses a ticketing system for secure verification of identities across an unsecured network. Primarily designed for client-server models, it emphasizes mutual authentication, enabling both the user and server to verify each other's identities reliably.

At the heart of Kerberos is the Key Distribution Center (KDC), a third-party entity responsible for managing the authentication process and ticket granting. The KDC is composed of two critical components:

1. **Authentication Server (AS)**: This component is responsible for authenticating users or entities in the network. Upon successful authentication, the AS issues a Ticket-Granting Ticket (TGT), which is encrypted and serves as proof of the user's identity for the session.
2. **Ticket Granting Server (TGS)**: Once a user has a TGT, they can request access to various services within the network. The TGS receives the TGT, verifies it, and if the user is authorized, issues a service-specific ticket. This ticket then allows the user to access the desired service without re-authenticating.

Kerberos Authentication Steps:

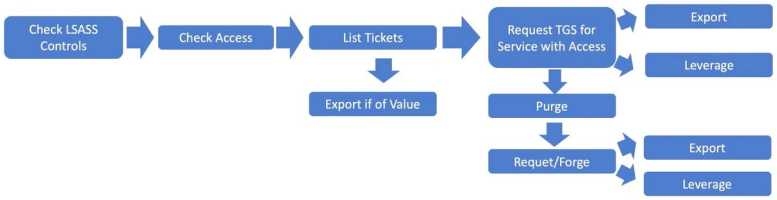
1. **KRB_AS_REQ**: Client sends an encrypted timestamp using the user's NTLM hash to the Authentication Server (AS) for mutual verification and requests a Ticket-Granting Ticket (TGT), prevent replay attacks, since the Timestamp value must be in the last minutes
2. **KRB_AS_REP**: Upon authentication, the AS issues a TGT containing a timestamp and session key, encrypted for the client, enabling future authenticated requests without the NTLM hash.
3. **KRB_TGS_REQ**: The client requests access to a service by sending the TGT and a service request (encrypted with the AS-provided session key) to the Ticket Granting Server (TGS), including the service identifier and supported encryption methods.
4. **KRB_TGS_REP**: The TGS issues a service ticket divided into two parts: one encrypted with a session key for the client (including a new session key for service access), and another for the service server, detailing the requester's privileges and session key, encrypted with the service account's NTLM hash.
5. **KRB_AP_REQ**: The client presents the service ticket to the service server to authenticate and access the service.
6. **KRB_AP_REP:** (Optional) The service server may send a response, encrypted with the session key, to establish secure communication.



Q3 - Pass-the-Ticket Attack: A Pass-the-Ticket (PtT) attack exploits the Kerberos ticketing system by stealing and reusing TGTs or service tickets, allowing attackers to impersonate legitimate users. This attack bypasses the need for passwords, leveraging stolen tickets for unauthorized access to resources.

How PtT Works:

1. **Initial Compromise**: The attacker gains access to a system (e.g., via phishing)
2. **LSASS Control**: Targeting the Local Security Authority Subsystem Service (LSASS) in Windows, which stores login credentials, is crucial for the attacker to access and manipulate credentials.
3. **Access Level Check**: The attacker evaluates their system access; administrative privileges are often necessary to exploit LSASS for Kerberos ticket extraction.
4. **Ticket Identification**: Kerberos tickets, especially Ticket Granting Tickets (TGTs), are identified from system memory for potential use.
5. **Ticket Assessment and Use**: Valuable tickets are either saved for later or immediately used. The attacker may also use a TGT to request service-specific tickets from the TGS.
6. **Ticket Management**: Depending on their value or potential detection risk, tickets are exported for later use, leveraged for immediate access, or deleted to cover tracks.
7. **Ticket Acquisition**: The attacker might forge new tickets or request additional ones from the TGS, using existing credentials or cryptographic methods.
8. **Utilization**: In the final step, the attacker either stockpiles these newly obtained tickets or uses them promptly for unauthorized access.



Detecting PtT Attacks: Detecting Pass-the-Ticket (PtT) Attacks: Effective detection strategies encompass monitoring for anomalies in access patterns and authentication requests, conducting thorough audits of Kerberos authentication logs to identify discrepancies, and keeping a vigilant eye on the lifespan of tickets to flag any usage beyond their intended validity period.

Defending Against Pass-the-Ticket (PtT) Attacks: Effective strategies for mitigating PtT attacks involve several layers of defense. Firstly, adjusting the lifespan of Ticket Granting Tickets (TGTs) and service tickets to a shorter duration, typically a 10-hour maximum with the option for a 7-day renewal, significantly reduces the opportunity window for attackers. Enhancing account security is another critical step; this can be achieved by implementing multifactor authentication (MFA) and restricting the proliferation of privileged accounts, thereby minimizing attack vectors. Proactive and continuous monitoring Furthermore, the deployment of Credential Guard enhances security through virtualization-based isolation of LSASS, safeguarding credentials within a virtualized environment (Isolates LSASS, Secures Kerberos Tickets, Thwarts Credential Theft).

Response Strategies: Reset Compromised Accounts, Double Reset KRBTGT, Analyze Access and Data Breach.

Understanding New Technology Lan Manager (NTLM) Authentication

Introduction

A domain represents a network of computers managed under a unified set of rules by an administrator. The Domain Controller (DC) enforces these rules, authenticates users, and stores their information. Microsoft's Active Directory (AD) is a suite of services facilitating network management, encompassing directory services via LDAP, DNS services, and notably, authentication services including Kerberos.

Domain validation acts as an intermediary, ensuring seamless coordination and data exchange between various departments.

Hash: A hash turns data into a fixed-length string. It's a **one-way process**, like making a smoothie you can't go back to the original ingredients. If you change anything in the data, the hash changes entirely. Hello” → Hash → “5d41402abc4basf4b9719d911017c592”

The NTLM Protocol New Technology LAN Manager (NTLM) is a challenge-response authentication protocol used to verify a user's identity without transmitting the password directly.

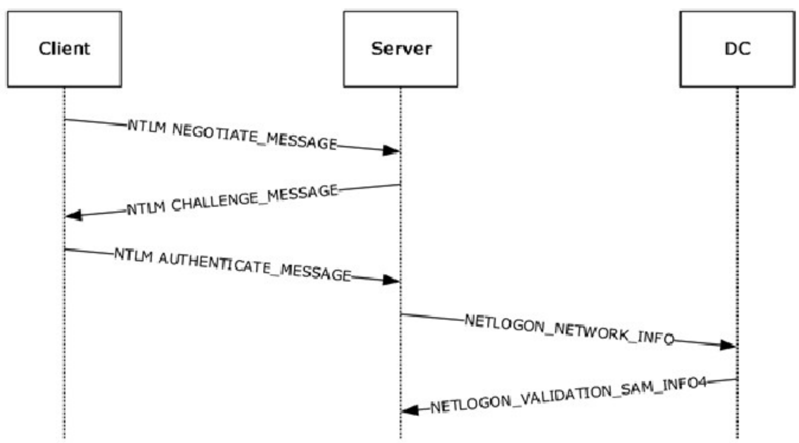
NTLM Authentication Steps:

1. **Negotiation**: A user initiates authentication to a remote machine, sending an NTLM_NEGOTIATE_MESSAGE.
2. **Challenge Request**: The server responds with a challenge, an NTLM_CHALLENGE.
3. **Challenge Response**: The user encrypts this challenge with their password's hash, replying with an NTLM_AUTHENTICATE message.
4. **Domain Controller Validation**: The server verifies this response with the Domain Controller via a NETLOGON request.
5. **Authentication and Session Key Exchange**: If validated, the DC attaches a SESSION key for encrypted communication, which the client can also independently calculate.

Examples for Clarity:

- A user accessing a shared folder sends an NTLM negotiation message.
- The server issues a challenge, which the user encrypts with their hashed password.
- The encrypted challenge is validated by the Domain Controller, enabling secure session key exchange.

**each client has its own private hash method



Q4 - NTLM Vulnerabilities and Relay Attack

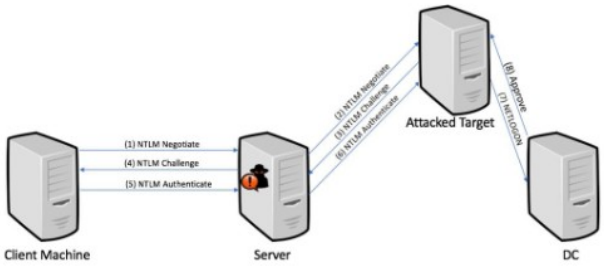
Key Vulnerabilities:

1. **Lack of Mutual Authentication**: Only the client authenticates to the server, not vice versa, allowing potential man-in-the-middle attacks.
2. **Use of Hashes for Storage**: Password hashes stored on servers can be targeted for cracking.
3. **Susceptibility to Pass-the-Hash and Brute Force Attacks**: Stolen hashes can be used for unauthorized access, and the absence of salting makes brute force attacks more feasible.
4. **NTLMv1's Known Weaknesses**: This outdated version lacks modern security features, making it vulnerable.
5. **Relay Attacks**: Attackers can intercept and relay authentication requests to gain unauthorized access.

NTLM Relay Attack Explained: An NTLM Relay Attack intercepts authentication requests between a client and a server. The attacker then relays these requests to access resources or services without needing the user's password. This type of attack exploits the trust relationship between the client and server, requiring network access and specialized relay software to manipulate and forward authentication data. This exploitation relies on: Lan access, ARP Spoofing, Compromised Network Devices, API Vulnerabilities and Relay Software.

Detecting NTLM Relay Attacks: Detection strategies include timestamp analysis for authentication timing anomalies, network traffic monitoring for unusual relay activities, and geolocation checks in the authentication process.

Defending Against NTLM Relay Attacks: Defense strategies encompass enforcing SMB Signing to authenticate and secure SMB communications, implementing network segmentation , "upgrade" to Kerberos authentication and applying the principle of least privilege to minimize exposure from compromised credentials.



Response Strategies: Reset Passwords, Isolate Affected Systems, Review logs.
History?

- NTLMv1NTLMv1 was introduced with Windows NT 3.1 in 1993.
- NTLMv2 was introduced with Windows NT 4.0 SP4. In 1998.

NTLMv1's fixed hash length makes it vulnerable to quick cracking.
NTLMv2 improves security with timestamps and variable-length challenges, yet it retains some NTLMv1 flaws due to a similar authentication process.

The protocol's one-way authentication diminishes security by potentially exposing users to misauthentication with fraudulent servers, making it an attractive target for attackers.
Understanding NTLM is essential because, in numerous Microsoft Windows environments, it functions as the fallback authentication protocol when Kerberos authentication is not feasible.