

CS203 Gene's Class - MITM Attack and Defense



Many bank websites have adopted the following anti-fraud defense. The first time a user registers at the bank's website, she enters her username and password as usual, and is given a choice between several pictures. The association between the username and the chosen picture is stored in the bank's database. In all subsequent sessions, the user types in her username and expects to be shown a picture. Unless she sees the picture she chose during her first session, she does not type in her password. This way, users avoid giving their passwords to fake websites.

A) Describe a man-in-the-middle attack that allows a fake website to show the user her chosen picture. (Assume that this is not the user's first session, i.e., she has already chosen the picture.)

B) Design a cookie-based defense for this anti-phishing scheme that prevents the man-in-the-middle attack you discovered in part A.

A) Man-in-the-Middle Attack to Show Chosen Picture

This defense aims to stop phishing by ensuring that the legitimate site shows a user-specific image *before* the password is entered. But it can be bypassed by a **man-in-the-middle (MITM)** phishing site as follows:

Attack Description:

1. **Victim visits a phishing site** (`fakebank.com`) and enters her **username**.
2. The phishing site **relays** the username to the real bank website (`realbank.com`) in the background.
3. The real bank, believing it's a genuine session, returns the **associated picture** for that username (e.g., "cat.jpg").
4. The phishing site displays the correct picture to the victim.
5. The victim is **fooled into thinking** the site is legitimate, so she enters her **password**.
6. The phishing site captures the password and may optionally forward it to the real site to complete login or hijack the session.

This works because the picture is associated only with the **username**, and there's **no additional authentication** at this stage. The bank serves the picture to anyone who asks with a valid username—even attackers.

B) Cookie-Based Defense

To prevent this attack, we need to make sure that only **the legitimate browser that previously registered** (or logged in) can retrieve the picture, not an attacker's relay server.

✓ Defense: Bind Image Display to a Client-Side Cookie

1. When the user **first logs in** and chooses the picture, the server sets a **secure, HttpOnly, origin-bound cookie** (e.g., `image_cookie`) in the user's browser.
2. For **subsequent sessions**, the bank **only shows the chosen picture** if the request includes both:
 - the correct **username**, and
 - the valid `image_cookie`.
3. If the cookie is missing or invalid (as in a phishing site or MITM), the **bank withholds the picture**, displays a warning, or refuses to proceed.
4. The cookie should:
 - Be tied to the origin (e.g., `https://realbank.com`) and **not sent to phishing sites**.
 - Be long-lived, but rotated occasionally for security.
 - Optionally be tied to a device fingerprint to reduce theft risk.

Why It Works:

- A **phishing site** cannot access cookies from `realbank.com` due to **same-origin policy**.
- Thus, when the attacker relays the username to the real bank, **the cookie is missing**, so the bank **refuses to show the picture**.
- The **victim does not see her picture**, realizes something is wrong, and does **not enter her password**.

Would you like a diagram to illustrate this defense visually?



Suppose that, from today onward, UCI is blocking all packets to/from your favorite streaming service using a firewall that works by checking source and destination IP addresses and discarding packets if either the source (of incoming) or destination (of outgoing) packets is in that streaming service. To survive for the remainder of the quarter, you must find a way to bypass this restriction. Explain how IPSec can help you to bypass the firewall and regain access to your favorite streaming services. Which IPSec mode (Tunnel or Transport) is more appropriate considering this goal and why



To **bypass UCI's IP-based firewall restriction** and access your favorite streaming service, you can use **IPSec**, a protocol suite that secures network traffic at the IP layer. Here's how:

How IPSec Helps Bypass the Firewall:

- The firewall at UCI blocks **all IP packets to/from the streaming service**, based on **source or destination IP addresses**.

- IPsec **encrypts the entire IP packet**, including the destination IP address (depending on the mode), making it invisible to the firewall.
- To bypass the firewall:
 1. You set up a **VPN connection** (Virtual Private Network) to a **remote VPN gateway/server** *outside* UCI's network.
 2. The VPN uses **IPsec Tunnel Mode**, which encapsulates and encrypts your original IP packet—including the streaming service's blocked IP.
 3. All the UCI firewall sees is **encrypted traffic between you and the VPN server**, not the true destination.
 4. The VPN server decrypts the packet and forwards it to the streaming service, and returns responses via the same secure tunnel.

Why Tunnel Mode is More Appropriate Than Transport Mode:

Feature	Tunnel Mode	Transport Mode
Encryption Scope	Encrypts entire original IP packet	Encrypts only the payload (not the IP header)
IP Header Visibility	Hides both source and destination IP addresses	Leaves destination IP visible
Common Use Case	Site-to-site or remote-access VPNs	End-to-end host security within trusted networks
Bypass Suitability	 Best for hiding traffic from firewalls	 Firewall can still see blocked destination IP

- **Tunnel Mode** wraps the whole IP packet (with the blocked IP address) inside a **new IP packet** addressed to the VPN server—so **UCI's firewall cannot detect or block** it based on IP addresses.
- **Transport Mode** would **not** help because the destination IP address of the streaming service remains visible to the firewall.

Final Answer:

- **Use IPsec in Tunnel Mode** to create a VPN to a remote gateway.
- This encrypts the entire IP packet, hides the streaming service's IP address, and **bypasses UCI's firewall** successfully.

Would you like a diagram or quick command-line example to help set this up?