Week 09:

**Control is - An action, device, procedure, or other measure that reduces risk by eliminating or preventing a security violation, by minimizing the harm it can cause, or by discovering and reporting it to enable corrective action.**

**CONTROL CLASSIFICATIONS**

Management controls

* Focus on security policies, planning, guidelines, and standards that influence the selection of operational and technical controls to reduce the risk of loss and to protect the organization’s mission.
* These controls refer to issues that management needs to address.

Operational controls

* Address the correct implementation and use of security policies and standards, ensuring consistency in security operations and correcting identified operational deficiencies.
* Primarily implemented by people rather than systems.
* Improve the security of a system or group of systems.

Technical controls

* Involve the correct use of hardware and software security capabilities in systems.
* These range from simple to complex measures that work together to secure critical and sensitive data, information, and IT systems functions.

**CONTROL CLASSES**

Supportive controls

* Pervasive, generic, underlying technical IT security capabilities that are interrelated with, and used by, many other controls.

Preventative controls

* Focus on preventing security breaches from occurring, by inhibiting attempts to violate security policies or exploit a vulnerability.

Detection and recovery controls

* Focus on the response to a security breach, by warning of violations or attempted violations of security policies or the identified exploit of a vulnerability and by providing means to restore the resulting lost computing resources.

**TECHNICAL CONTROLS-TCP/IP SECURITY SOLUTION**

A number of approaches to providw Internet security

A close-up of a computer

Description automatically generated

Network Level Security: This diagram indicates that security measures such as IPsec operate at the network layer, encapsulating the TCP layer and application protocols like HTTP, FTP, and SMTP.

Transport Level Security: It shows that security protocols like SSL (Secure Sockets Layer) or TLS (Transport Layer Security) are implemented at the transport layer, securing the data transmission for the application layer protocols listed above.

Application-Level Security: The final diagram presents application layer security mechanisms such as S/MIME for email encryption and Kerberos for network authentication, which are applied directly to specific application protocols.

**APPLICATION LAYER PROTOCOL**

**EMAIL SECURITY: MIME AND S/MIME**

**Multipurpose Internet Mail Extension -** Simple heading with To, From, Subject. *ASCII text format.* Provides several new header Fields that define information about the body of the message.

**Secure/Multipurpose Internet Mail Extension -** Security enhancement to the MIME Internet e-mail format. Based on technology from RSA Data Security. Provides the ability to sign and/or encrypt e-mail messages.

A diagram of a data analysis

Description automatically generated with medium confidence

Enveloped data - The "encrypted content and associated keys" portion implies that the data itself is encrypted, and there are cryptographic keys related to this process.

Signed data - "Encoded message + signed digest" indicates that the message is first encoded and then a digest (hash) of the message is created and signed with the sender's private key.

Clear signed data - The "cleartext message + encoded signed digest" means that the message itself isn't encrypted and is readable, but it comes with a signed hash to verify the sender and ensure the message hasn't changed during transit.

Signed and enveloped data - This is a combination of the first two functions, providing both encryption and digital signature. The message is encrypted for confidentiality and signed to ensure integrity and authenticity.

Process involves wrapping the signed message within an encrypted layer, offering a comprehensive security approach.

**PRETTY GOOD PRIVACY (PGP) CRYPTOGRAPHY**

- PGP is a standard for electronic-mail encryption and digital signatures.

- It utilizes a Public-Private Keys (PPK) method.

- Users can sign each other's public keys, enhancing trust in their validity.

- Signing someone's public key acts as an introduction, allowing users to trust individuals based on the introducer's trust.

- PGP is commonly used for encrypting documents shared via email over the open Internet.

- Contrary to S/MIME and OpenPGP, PGP employs its encryption techniques and digital signature handling methods.

To prevent DNS Hijacking and DNS Pharming, **DNS SECURITY** (DNSSEC) is deployed to ensure:

**Authenticity of DNS Answer Origin**: Verifies that the DNS response originates from a genuine and authorized source.

**Integrity of Reply**: Ensures that DNS responses remain unaltered and free from tampering during transmission.

**Authenticity of Denial of Existence**: Validates assertions of non-existence of DNS records, preventing spoofed or false negative responses.

It also introduces challenges such as increased packet sizes, exceeding the standard 512-byte limit of UDP packets.

Leads to additional load on DNS servers. Uses public key crypto to sign responses.

**SECURE SHELL (SSH)** - A protocol designed for secure network communications, prioritizing simplicity, and cost-effectiveness in implementation.

Initially, SSH1 aimed to provide a secure remote login facility, replacing insecure protocols like Telnet.

SSH offers a broader client/server capability, enabling functions like file transfer and email over secure channels.

SSH client and server applications are readily accessible across various operating systems.

SSH2 addresses security vulnerabilities present in the original SSH scheme, enhancing overall security for users and systems.

**SSH PROTOCOL PACKET EXCHANGE**

A diagram of a computer network

Description automatically generated

TCP Connection Establishment:

- The process begins with the client establishing a Transmission Control Protocol (TCP) connection to the server.

- This TCP connection is not encrypted; it's a standard internet protocol that provides reliable, ordered, and error-checked delivery of a stream of data between applications.\

- The establishment of this TCP connection is not part of the SSH Transport Layer Protocol; it's a prerequisite that allows the SSH protocol to operate over a reliable channel.

SSH Secure Connection:

- Once the TCP connection is established, the client initiates the SSH protocol which involves negotiating encryption algorithms, exchanging cryptographic keys, and authenticating the server and possibly the client.

- After successfully completing this handshake and establishing the encrypted SSH tunnel, the client and server can exchange data securely.

- The data packets sent back and forth are encapsulated within the SSH protocol, which is tunnelled over the TCP connection.

- SSH ensures that the data exchange is confidential (due to encryption), authenticated (the server and possibly the client prove their identities), and maintains integrity (it prevents data from being altered in transit).

**SSH PROTOCOL STACK**

A diagram of a computer

Description automatically generated

the stages involved in the SSH (Secure Shell) protocol communication stack during a session establishment between a client and a server.

**Establish TCP Connection:** This is the first step where the client initiates a TCP connection with the server, providing a reliable stream of bytes between the two.

**Identification String Exchange:** Once the TCP connection is established, the client and the server exchange identification strings. This usually includes information about the SSH protocol version and the software version. This is to ensure that both client and server are using compatible versions of SSH.

**Algorithm Negotiation:** The next step involves the negotiation of the cryptographic algorithms that will be used for the session. This can include the encryption, key exchange, and MAC (Message Authentication Code) algorithms. Both the client and the server need to agree on a set of algorithms to use.

**Key Exchange:** This stage is where the actual key exchange occurs using the negotiated key exchange algorithm. The key exchange process is essential for establishing a secure channel because it allows both parties to have the same cryptographic keys for encryption and decryption without sending the keys across the network.

**End of Key Exchange:** At this point, a message (SSH\_MSG\_NEWKEYS) is sent by both the client and the server to signal the end of the key exchange process. After this message is sent, all subsequent communication is encrypted using the agreed algorithms and keys.

**Service Request:** The client sends a service request (SSH\_MSG\_SERVICE\_REQUEST) to the server requesting a specific type of service (e.g., shell access, file transfer). The server then responds whether it can start the requested service. After the server acknowledges the service request, the client can begin the authentication process to access the requested service.

The diagram probably represents the sequential nature of these steps and emphasizes the back-and-forth communication.

**TLS PROTOCOL STACK**

TLS is designed to make use of TCP to provide a reliable end-to-end secure service.

* Three higher-layer protocols are defined as part of TLS:

The Handshake Protocol.

The Change Cipher Spec Protocol.

The Alert Protocol.

These TLS specific protocols are used in the management of TLS exchanges.

A fourth protocol, the Heartbeat Protocol, is defined in a separate RFC.

**TLS CONCEPTS**

✺ TLS Session:

Created by the Handshake Protocol

define a set of cryptographic parameters.

Used to avoid the expensive negotiation of new security parameters for each connection.

✺ \_TLS Connection:

A transport layer protocol that provides a suitable type of service

Peer-to-peer relationships

Every connection is associated with one session.

**TLS HANDSHAKE MESSAGES**

The most complicated aspect of TLS.

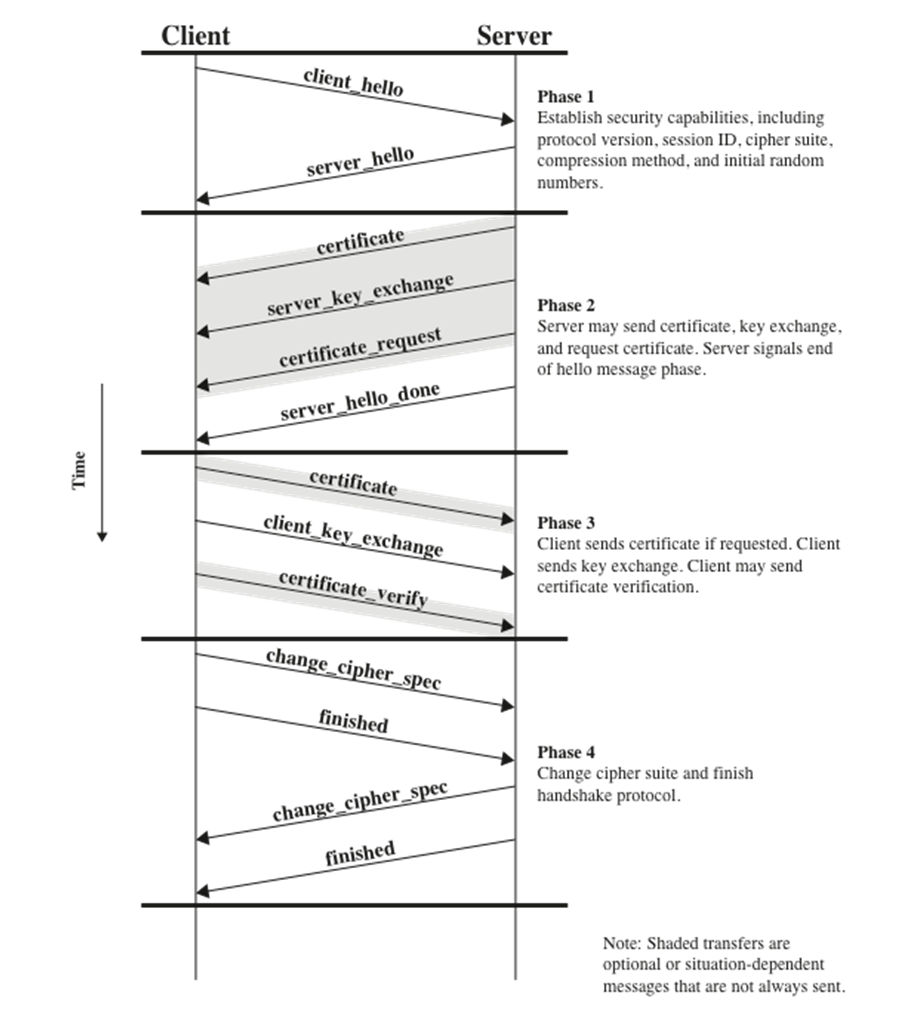
It occurs before any application data is sent over the connection.

The handshake allows the server and client to authenticate each other, negotiate encryption and MAC (Message Authentication Code) algorithms, and agree on cryptographic keys for secure communication.

The handshake involves a series of messages exchanged between the client and server.

The exchange unfolds in four distinct phases, each serving a specific purpose in establishing a secure TLS connection.

**HANDSHAKE PROTOCOL ACTION**



TLS (Transport Layer Security) handshake process.

**Client Hello:** The client initiates the handshake by sending a "hello" message to the server. This message will include the client's TLS version, the cipher suites supported by the client, and a string of random bytes known as the "client random."

**Server Hello:** In response, the server sends a "hello" message back to the client. This message contains the server's TLS version, the cipher suite chosen by the server from the list provided by the client, and another string of random bytes known as the "server random."

**Certificate:** The server sends its digital certificate to the client. The certificate contains the server's public key and is used to establish the server's identity.

**Server Key Exchange (if necessary):** If the agreed-upon cipher suite requires additional information for the setup of a secure connection, the server will send a key exchange message.

**Certificate Request (optional):** The server can request a certificate from the client, for mutual TLS authentication.

**Server Hello Done:** This message indicates that the server has finished sending messages to initiate the TLS handshake and is now waiting for the client's response.

**Certificate (optional):** The client sends a certificate to the server if requested.

**Client Key Exchange:** The client sends a key exchange message, which contains the pre-master secret encrypted with the server's public key (from the server's certificate).

**Certificate Verify (optional):** If the client sent a certificate, it must also send a message to verify the certificate.

**Change Cipher Spec:** Both the client and the server send a message to indicate that subsequent messages will be encrypted using the negotiated cipher suite and keys.

**Finished:** Both the client and the server send a message confirming that the handshake is complete and that the secure encrypted connection has been established.

The "Finished" messages are encrypted and contain a hash of all the transmitted handshake messages. If the hashes match on both ends, it confirms that the handshake was not tampered with and that both the client and server have the same shared secret, allowing for secure communication.

The TLS handshake is a complex process designed to secure communications so that third parties cannot eavesdrop or tamper with the messages. It uses asymmetric cryptography for the initial key exchange and then switches to symmetric encryption for communication after the handshake is complete, providing a blend of security and performance.

**HTTPS (HTTP OVER TSL)**

Combination of HTTP and TLS (RFC 2818, HTTP Over TLS) to implement secure communication between a Web browser and a Web server.

Built into all modern Web browsers.

URL addresses begin with https://

Agent acting as the HTTP client also acts as the TLS client.

Closure of an HTTPS connection requires that TLS close the connection with the peer TLS entity on the remote side.

**APPLICATIONS OF IPSEC**

IPsec offers the capability to secure communications across local area networks (LANs), private and public wide area networks (WANs), and the Internet.

Examples of Applications:

- Secure branch office connectivity over the Internet.

- Secure remote access over the Internet.

- Establishing extranet and intranet connectivity with partners.

- Enhancing electronic commerce security.

IPsec's main feature is its ability to encrypt and/or authenticate all traffic at the IP level. This means that all distributed applications, including remote logon, client/server, email, file transfer, and web access, can be secured using IPsec.

**IPSEC SERVICES**

IPsec offers security services at the IP layer, allowing a system to:

- Select required security protocols.

- Determine the algorithms for the desired services.

- Establish necessary cryptographic keys for providing the requested services.

RFC 4301 Services:

- Access control.

- Connectionless integrity.

- Data origin authentication.

- Rejection of replayed packets (integrity).

- Confidentiality (encryption/confidentiality).

**BENEFITS OF IPSEC**

When IPsec is implemented in a firewall or router, it provides strong security that can be applied to all traffic crossing the perimeter.

Traffic within a company or workgroup does not incur the overhead of security-related processing.

IPsec is below the transport layer (TCP, UDP) and so is transparent to applications.

There is no need to train users on security mechanisms.

This is useful for offsite workers and for setting up a secure virtual subnetwork within an organisation for sensitive applications.

**THE SCOPE OF IPSEC**

Provides two main functions:

* A combined authentication/encryption function called Encapsulating Security Payload (ESP)
* Key exchange function

Also, an authentication-only function, implemented using an Authentication Header (AH) - Because message authentication is provided by ESP, the use of AH is included in IPsecv3 for backward compatibility but should not be used in new applications

VPNs want both authentication and encryption.

**TRANSPORT MODE**

Provides protection for upper layer protocols.

* Examples: TCP or UDP segment or an ICMP packet

Used for end-to-end communication between two hosts.

ESP in transport mode encrypts and optionally authenticates the IP payload but not the IP header.

AH in transport mode authenticates the IP payload and selected portions of the IP head.

**TUNNEL MODE**

Provides protection to the entire IP packet.

* Used when one or both ends of a security association (SA) are a security gateway.

ESP in tunnel mode encrypts and optionally authenticates the entire inner IP packet, including the inner IP header.

AH in tunnel mode authenticates the entire inner IP packet and selected portions of the outer IP header.

**IPSEC: TUNNEL MODE FORMAT**

Tunnel Mode Components: In IPsec, tunnel mode utilizes Encapsulating Security Payload (ESP), which combines authentication and encryption functions, along with a key exchange function.

VPN Requirement: For Virtual Private Networks (VPNs), both authentication and encryption are typically desired. This is crucial to prevent unauthorized users from accessing the VPN and ensure that messages sent over the VPN remain secure from eavesdroppers on the Internet.