NIS Assignment

Task 2: You are required to code and report on an application that **can encrypt a message with only a symmetric key**. You may need to create two programs (one to send messages, and the other to receive messages). The message sent must not be susceptible to modification, replay or traffic analysis attacks. List the security features that you implement to prevent against these attacks, as well as any additional features that may protect against other attacks. Also, describe how these features ensure security.

Program Layout

Language: Golang
API libraries: "crypto/tls"

Program Assumptions:

In order to ensure security and prevent replay attacks the docker image has been modified slightly from a typical Debian image. Our certificate has been manually added to the trusted certificates for the image within the /usr/local/share/ca-certificates directory. If you check into this directory you will note the server.crt is manually installed within this directory. This is to ensure our certificate is temporarily trusted by the server when the client connects for demonstration purposes. This allows us to bypass having to have our certificate signed by the Distinguished Names of acceptable Certification Authorities (CAs) (trusted CAs) that would create a certified certificate chain deeming the certificate as valid (since this approach is more cumbersome and involved). In addition, it enables our local address 127.0.0.1 to temporarily have the ability to create and terminate “trusted” Transport Layer Security (TLS) connections, virtually enabling https connections to localhost.

Implementation Method: Transport Layer Security (TLS) / Secure Socket Layer (SSL)

TLS is a transport layer security protocol that enables the secure creation of an SSL connection between communicating parties. The establishment of the secure connection must happen using asymmetric security techniques such as RSA, however message passing once the connection has been established is solely done via AES-256 which is a symmetric means. The mix of both symmetric and asymmetric techniques ensures that key sharing and initial contact is secure thanks to public key cryptography, and that security and speed is achieved later using AES which is much faster.

How TLS avoids Replay and Man In the Middle Attacks (MITM)

TLS has the ability to secure the transport of application layer messages, ensuring data confidentiality and integrity as messages are protected against modification however this is to a certain extent. The use of Message Authentication Code or MAC is computed from the MAC secret and sequence number ensuring the prevention of replay or modification attacks. Given an attacker attempting to replay the initial client hello message, the server would respond with a corresponding server hello but with a different random value that in effect changes the rest of the key exchange. If a client replays a message under a different SSL connection, then the server has to have built in logic to handle the idempotent request. TLS will therefore only protect against replay and MITM within a given session, however if the client is compromised by malicious intent it can create several individual TLS connections and perform replay through those separate connections. However this is okay given the assignment and is considered as an assumption

Features Implemented To Ensure Security

The application, with send and receive programs, used cryptography API libraries with AES encryption on Golang and Transport Layer Security (TLS) to implement security features that ensure security of the symmetric keys, their storage and distribution, and message transfers between sending and receiving parties using a set of security services. The security services implemented as features in the application entail the following.

Authentication and Authenticity

Certificate authentication library on Golang was used to implement authentication feature (X509 certificates) for sender and receiver programs to uniquely identify each other. This entailed requesting of Signed Certifications between sender and receiver parties for verification before key/message are exchanged and for loading public-private key pair to be used from encoded data. This is also supplemented by the use of session states and key.

Data Access Control

The generated keys are stored/retrieved safely with the use of trusted certificates for loading key pair and access is restricted to the authenticated user to ensure that no unrecognized access, such as read, write and execute, is made to illegally obtain the keys or illegally execute a send/receive program. This is crucial because if the private and session keys are accessed, they can be used to perform misuse of sending or receiving program, resources and data breach.

Non-Repudiation - undeniable send or receive

Signing of the symmetric keys is performed which ensures that receiver X can verify that it was sender Y who sent the message and sender Y can verify that receiver X did receive a message from Y proven with a trusted certificate, signatures and the use of a session key between these communicating parties and thus either party cannot deny that certain actions were not performed by them.

Confidentiality and Integrity

The incorporation of TLS and SSL on the connection setup between sender and receiver ensures that the connection is only visible between these communicating parties and that no third party can intercept connection to read and modify data or keys on transit. This protects the privacy of the connection and data from possible security attacks and network analysis. Integrity is achieved with a scheme that uses AES-256 hashing algorithm to ensure that data or key modification do not occur.

References

[1] *Package crypto tls*, Go Programming Language, https://golang.org/pkg/crypto/tls/