Question Number 12

TLS 1.3

As shown in CIA-1, using the openssl dumps with a local client-server compare and contrast TLS 1.2 and TLS 1.3. Submit all necessary files and a script to verify your approach.

Now write a note in your notebook highlighting how TLS 1.3 improves upon TLS 1.2.

Solution.

Setup

- Create private key and certificate by following the instruction mentioned in `CreateKey.md` file
- Run the Server and Client in differnet terminals for `TLS1.2` and `TLS1.3` by following instruction in `handshake.md` file
- Observe the output dumped in terminal
- Instead doing of all these instruction run script file `run.sh` following instruction from `README.md` file and observer the output which is dumped in 4 different spawn gnome-terminal.

Comparison of TLS 1.2 and TLS 1.3 Using OpenSSL

1. Cipher Suites

- **TLS 1.2:**
  - **Output**
    - Server → Shared ciphers:ECDHE-ECDSA-AES256-GCM-...
    - Client → Cipher: ECDHE-RSA-AES256-GCM-SHA384
  - Offers a broad range of cipher suites, including `ECDHE-ECDSA-AES256-GCM-SHA384`, `ECDHE-RSA-AES256-GCM-SHA384`, and `DHE-RSA-AES256-GCM-SHA384`, among others.
  - The cipher suite used in this session is `ECDHE-RSA-AES256-GCM-SHA384`.
- **TLS 1.3:**
  - Simplifies the cipher suites by reducing the number of options, focusing on stronger and more secure algorithms. The shared cipher suites are `TLS_AES_256_GCM_SHA384`, `TLS_CHACHA20_POLY1305_SHA256`, and `TLS_AES_128_GCM_SHA256`.
  - The cipher suite used in this session is `TLS_AES_256_GCM_SHA384`.

2. Handshake Process

- **TLS 1.2:**
  - The handshake process is more complex and involves multiple round-trips between the client and server.

  - Uses different key exchange mechanisms, including ECDHE and DHE.
  - The session output indicates that the server supports secure renegotiation.

- **TLS 1.3:**
  - Simplifies the handshake process by reducing it to one round-trip (1-RTT), improving performance and security.
  - Eliminates support for certain older cryptographic algorithms, focusing on a smaller set of secure options.
  - Like TLS 1.2, the session indicates support for secure renegotiation, but TLS 1.3 inherently avoids the pitfalls of renegotiation attacks by streamlining the process.

3. Supported Groups and Algorithms

- **TLS 1.2:**
  - Supports a wide variety of elliptic curve groups (`x25519`, `secp256r1`, `x448`, etc.) and signature algorithms.
- **TLS 1.3:**
  - Expands support for Diffie-Hellman groups (`ffdhe2048`, `ffdhe3072`, etc.), reflecting a shift toward more modern and secure cryptographic primitives.
  - Continues support for elliptic curve groups but omits older, less secure options.

4. Security Enhancements

- **TLS 1.2:**
  - Offers optional support for forward secrecy if specific cipher suites (like ECDHE) are used.
- **TLS 1.3:**
  - Enforces forward secrecy by default, as all cipher suites supported by TLS 1.3 are forward secure.
  - Removes support for older, less secure features like RSA key exchange and static DH, further enhancing security.

5. Session Resumption

- **TLS 1.2:**
  - Relies on session tickets and session IDs for session resumption, which involves some risks like ticket reuse.
- **TLS 1.3:**
  - Uses session tickets in a more secure manner and introduces Pre-Shared Key (PSK) modes for session resumption, reducing the chance of ticket reuse attacks.

6. Signature Algorithms

- **TLS 1.2:**
  - Supports a wide range of signature algorithms, including RSA, DSA, and ECDSA, with different hash functions (e.g., SHA256, SHA384).
- **TLS 1.3:**
  - Consolidates the list of supported signature algorithms, focusing on those that are more secure, such as RSA-PSS and ECDSA, and removing support for older, weaker algorithms.

7. Backward Compatibility and Transition

- **TLS 1.2:**
  - Remains widely used and is necessary for compatibility with legacy systems and clients that do not support TLS 1.3.
- **TLS 1.3:**
  - Designed to be more secure and efficient, though its adoption requires updates to client and server implementations. It is backward compatible with TLS 1.2 but offers a clear advantage in terms of security and performance.

8. Certificate and Verification

- Both TLS 1.2 and TLS 1.3 rely on the same certificate infrastructure. However, the outputs provided indicate a self-signed certificate being used, leading to verification errors in both cases (`verify return:1`, `Verify return code: 18`).

Conclusion

TLS 1.3 offers significant improvements over TLS 1.2 in terms of security, performance, and simplicity. By reducing the number of cipher suites, streamlining the handshake process, and enforcing forward secrecy by default, TLS 1.3 addresses many of the shortcomings of TLS 1.2. However, TLS 1.2 remains essential for backward compatibility with systems that have not yet upgraded to support TLS 1.3.

Note that "how TLS 1.3 improves upon TLS 1.2." is written in Notebook.