REPORT

[Assignment 3 : TLS 1.3 Handshake 조사]

과 목 명	정보보호론
교 수 명	김 효 승
학 번	20237107
작 성 자	하 태 영
제 출 일	2025.05.23

한림대학교

1. Handshake 흐름

TLS 1.3에서 1-RTT 및 PSK Handshake는 두 개의 뚜렷한 단계로 나뉩니다.

첫 번째는 키 교환 단계로, 이 단계에서는 클라이언트와 서버가 Hello 메시지를 교환하여 서로 지원하는 다양한 암호학적 옵션을 알리고, 선택된 파라미터를 사용해 키 교환 자료를 생성합니다.

두 번째는 인증 단계로, 이 단계에서는 클라이언트와 서버가 CertificateVerify 및 Finished 메시지를 교환하여, 장기 비대칭(또는 대칭) 값을 이용해 서로를 인증합니다.

참고문헌 : https://eprint.iacr.org/2020/1044.pdf (p9)

2. Cipher Suite 협상 방식

ClientHello

클라이언트는 ClientHello 메시지를 보내는 것으로 시작하는데, 이 메시지에는 rc(임의로 선택된 256비트 난수 값)와 버전 및 알고리즘 협상 정보가 포함되어 있습니다.

ServerHello

키 교환 단계의 다음 메시지는 ServerHello(SH)입니다. CH와 마찬가지로, 서버도 임의로 256비트 난수 값 rs를 생성합니다. 서버는 클라이언트가 제안한 다양한 알고리즘과 파라미터 중에서 하나를 선택해 그 선택 결과를 응답합니다.

참고문헌 : https://eprint.iacr.org/2020/1044.pdf (p9, p11)

3. 키 교환 및 인증 방식

- 키 교환

ClientHello

클라이언트는 ClientHello 메시지를 보내는 것으로 시작하며, 이 메시지에는 rc(임의로 생성된 256비트 난수 값), 버전 및 알고리즘 협상 정보가 포함되어 있습니다.
ClientHello에 첨부된 KeyShare(CKS) 확장에는 키 교환을 위한 공개키 값들이 포함되어 있습니다.
추가적인 확장들은 더 다양한 알고리즘 및 파라미터 협상을 위해 사용됩니다.
만약 클라이언트와 서버 사이에 사전 공유 비밀이 이미 존재한다면(이전 핸드셰이크에서 생성되었거나 별도의 방법으로 공유된 경우), 클라이언트는 PreSharedKey(CPSK) 확장을 포함할 수 있습니다.
이 확장은 클라이언트가 지원하는 핸드셰이크 모드(예: PSK 또는 PSK-(EC)DHE)와, 해당 PSK들과 매핑되는 사전 공유 대칭 식별자 목록을 나타냅니다.
CPSK가 포함된 경우, 클라이언트는 목록의 각 PSK에 대해 바인더 키 BK를 계산하고, 각 바인더 키와 현재 CH 메시지(바인더 값 자체는 제외)의 해시를 사용해 바인더 값을 계산합니다
: 바인더 ← HMAC(fKB, H(CH⁺))
이 바인더 값은 각 PSK에 대해 CPSK 메시지에 포함되며, 서버가 이를 확인합니다.
마지막으로, 클라이언트가 이 사전 공유 비밀을 사용해 0-RTT(0-라운드 트립 타임) 데이터를 전송하고자 한다면, EarlyDataIndication 확장을 보낼 수 있습니다.
이 확장은 서버에게, 클라이언트가 CPSK에 명시된 첫 번째 사전 공유 비밀로부터 초기 트래픽 시크릿(ETS)과 초기 exporter master secret(EEMS)을 도출해, 서버의 응답을 기다리지 않고 암호화된 데이터를 전송할 것임을 알립니다.

ServerHello

키 교환 단계의 다음 메시지는 ServerHello(SH)입니다. CH와 마찬가지로, 서버도 임의로 256비트 난수 값 rs를 생성합니다.
서버는 클라이언트가 제안한 다양한 알고리즘과 파라미터 중에서 하나를 선택해 그 선택 결과를 응답합니다.
CPSK가 포함된 경우, 서버는 PSK 기반 handshake를 수락할지 결정합니다.
만약 수락한다면, 선택된 PSK와 연관된 사전 공유 키 식별자를 PreSharedKey(SPSK) 확장에 담아 보냅니다.
서버가 (EC)DHE 모드를 선택했거나 PSK 사용을 거부했다면, 서버는 자신의 (EC)DHE 키 공유 값 Y = g^y를 생성해 KeyShare(SKS) 확장에 담아 보냅니다.
이 시점에서 서버는 클라이언트 handshake 트래픽 시크릿(CHTS)과 서버 handshake 트래픽 시크릿(SHTS) 값을 계산할 수 있으며, 이를 통해 클라이언트/서버 핸드셰이크 트래픽 키(tk_chs, tk_shs)를 도출합니다.
tk_chs와 tk_shs는 핸드셰이크 비밀(HS)이 사용 가능해지는 시점에 동시에 도출됩니다(실제로는 tk_chs가 약간 먼저 도출될 수 있습니다).
이제 서버는 tk_shs로 모든 핸드셰이크 메시지를 암호화하기 시작하며,
서버 handshake 트래픽 키가 필요하지 않은 추가 확장 메시지는 EncryptedExtensions(EE) 메시지로 암호화되어 전송됩니다.

3. 키 교환 및 인증 방식

- 인증 방식

1-RTT 핸드셰이크에서의 인증

서버는 CertificateRequest(CR) 메시지를 보내 클라이언트에 공개키 기반 인증을 요청할 수 있습니다.
서버는 자신의 장기 공개키를 이용해 클라이언트에게 인증합니다.
여기서 서버는 ServerCertificate(SCRT) 메시지로 자신의 인증서(공개키 포함)를 전송합니다.
서버는 세션 해시(프로토콜 진행 중 모든 메시지의 해시)를 서명해 ServerCertificateVerify 인증값을 계산한 뒤,
이를 ServerCertificateVerify 메시지로 클라이언트에 전송합니다.

서버 검증

서버는 클라이언트의 최종 MAC(SF) 및 선택적으로 서명(SCV) 메시지를 검증합니다.

핸드셰이크 완료

이 시점에서 양측 모두 재접속용 마스터 시크릿(RMS) 값을 도출할 수 있습니다. 이 값은 향후 세션 재개를 위한 사전 공유 키로 사용할 수 있습니다.
양측 모두 이제 클라이언트 애플리케이션 트래픽 키(tk_capp)를 도출해, 해당 키로 암호화된 애플리케이션 데이터 통신을 시작할 수 있습니다.

참고문헌 : https://eprint.iacr.org/2020/1044.pdf (p9, p11, p12, p13)

4. Key material 생성 방식

서버 키 확인 및 키 도출

모든 핸드셰이크 모드에서, 서버가 클라이언트에 보내는 마지막 메시지는 ServerFinished(SF) 메시지입니다.
서버는 먼저 SHTS로부터 서버 피니시드 키 fkS를 도출하고, 세션 해시를 이용해 MAC 태그 SF를 계산합니다.
이 메시지는 tk_shs로 암호화되어 전송되며, 암호화된 데이터를 클라이언트에 보냅니다.
이 시점에서 서버는 클라이언트 애플리케이션 트래픽 시크릿(CATS), 서버 애플리케이션 트래픽 시크릿(SATS), 익스포터 마스터 시크릿(EMS) 등
애플리케이션 트래픽 키와 기타 키들을 도출할 수 있습니다.
이제 서버가 클라이언트 애플리케이션 트래픽 키 tk_sapp을 도출한 뒤, 해당 키로 암호화된 애플리케이션 데이터를 전송할 수 있습니다.

클라이언트 검증, 인증, 키 확인 및 키 도출

클라이언트는 이러한 메시지를 수신한 후, 서명 SCV(1-RTT 모드의 경우)와 MAC SF가 올바른지 확인합니다.
서버가 클라이언트 인증을 요청한 경우, 클라이언트는 ClientCertificate(CCRT) 메시지로 자신의 디지털 인증서(공개키 포함)를 전송합니다.
이후 클라이언트는 세션 해시를 서명해 자신의 CertificateVerify 값 CCV를 계산하고, 이를 CCV 메시지로 서버에 전송합니다.
마지막으로 클라이언트는 CHTS로부터 클라이언트 피니시드 키 fkC를 도출해 세션 해시에 대한 MAC 태그 CF를 계산하고, 이를 서버에 보냅니다.

참고문헌 : https://eprint.iacr.org/2020/1044.pdf (p12, p13)

5. TLS 1.2와 비교

특징	TLS 1.2	TLS 1.3
핸드셰이크 프로세스	2 회 왕복 시간 (2-RTT)	1 회 왕복 시간 (1-RTT)
암호 스위트	약한 암호화 체계를 포함해 광범위한 암호화 체계를 지원합니다.	AEAD(연관된 데이터를 사용한 인증된 암호화) 암호 그룹만 허용되어 보안이 더욱 강화됩니다.
키 교환	RSA, DHE, ECDHE 가 지원됩니다.	ECDHE(Elliptic Curve Diffie-Hellman Ephemeral)만 지원됩니다.
전방 비밀 유지	선택 과목.	의무적 비밀 유지와 사전 비밀 유지는 기본적으로 시행됩니다.
암호화 알고리즘	RC4, CBC 와 같은 오래된 알고리즘이 포함되어 있습니다.	AES-GCM, ChaCha20-Poly1305 와 같은 더 강력한 알고리즘이 필요합니다.
세션 재개	재개를 위해 세션 ID 또는 세션 티켓을 사용합니다.	0-RTT 데이터로 세션 재개를 지원하여 지연 시간을 줄입니다.
보안 개선	BEAST, POODLE, Heartbleed 와 같은 공격에 취약합니다.	암호화된 핸드셰이크 메시지로 보안을 강화하고 안전하지 않은 프로토콜을 제거합니다.
인증서 검증	인증서 검증을 지원하지만 특정 핸드셰이크 세부 정보가 노출될 수 있습니다.	핸드셰이크 메시지가 암호화되므로 보안이 강화되었습니다.
O-RTT 데이터	지원되지 않습니다.	핸드셰이크 중에 데이터 전송을 지원 및 허용했습니다(하지만 재생 공격에 취약함).
성능	2-RTT 핸드셰이크와 추가 처리 단계로 인해 속도가 느려집니다.	간소화된 핸드셰이크와 최적화된 키 교환으로 인해 속도가 빨라졌습니다.

참고문헌 : https://www.encryptionconsulting.com/tls-1-2-and-tls-1-3/