인증서 압축을 활용한 TLS 확장 프로토콜의 적용 및 분석

이현우, 김영현, 조은상, 권태경
서울대학교

hwlee2014@mmlab.snu.ac.kr, young146777@snu.ac.kr, escho@mmlab.snu.ac.kr, tkkwon98@gmail.com

An Application and Analysis on TLS extension with Compressed Certificate

Hyunwoo Lee, Younghyun Kim, Eunsang Cho, Taekyoung Kwon
Seoul National University

요 약

본 논문은 TLS 핸드쉐이크에서 통신 상대를 인증하기 위한 인증서를 압축하는 TLS 확장의 적용 가능성을 살펴보았다. 대규모 업체가 사용하는 인증서는 크기가 19K 에 달하는 경우도 있기 때문에 압축된 인증서를 활용하면 서버는 트래픽 부하를 줄일 수 있다. 추가적으로 압축 알고리즘의 파라미터까지 협의한다면 저성능 기기에서도 사용할 수 있다.

Ⅰ. 서 론

인터넷에서의 암호화가 점차 증가하고 있다. [1]에 따르면, 웹에서 HTTPS 트래픽의 비율은 전체 트래픽 대비 50%를 넘어섰다. 이는 인터넷의 보안성과 개인정보에 대한 관심에서 비롯된 것이다.

하지만 보안성과 개인정보보호가 대가없이 주어진 것은 아니다[2]. 인터넷은 TLS 로 인해 트래픽 증가, 서버의 관리 부하 증가, 지연시간 상승 등의 단점을 안게 되었다. 또한 사물인터넷의 흐름에 따라 도입된 저사양 기기들의 부족한 연산 능력으로 말미암아 암호화 자체가 부하가 되기도 한다. 따라서 보안성을 보장하면서 부하를 줄이고 저사양 기기까지 포괄하기 위한 노력이 필요하다.

TLS 핸드쉐이크에서는 인증서가 많은 트래픽 부하를 발생시킨다. 인증서들의 크기는 대개 1KB 가 넘으며, 큰 것은 19KB 에 달하기도 한다. 이러한 부담을 줄이기 위해 최근 IETF 의 TLS WG 에서는 TLS Certificate Compression (이하 TLS-CC) 기고문[3]이 제출되기도 하였다.

본 논문은 위 필요성에 입각하여 TLS-CC 를 검토하고 적용, 분석해보고자 한다. 이를 위해 특정 시기에 인터넷에서 발견되는 인증서들의 크기와 압축 알고리즘을 적용했을 때의 크기 현황을 분석하고, 저사양 기기로의 도입 가능 여부와 이에 따른 이슈를 제기하였으며, 구현 및 실험을 진행하였다.

Ⅱ. 배경 이론

1. 전송 계층 보안 (Transport Layer Security, TLS)

TLS 란 RFC 5246[4]에 정의되어 있는 암호화 프로토콜로써 통신 상대에 대한 인증과 통신 상대와의 종단간 암호화, 그리고 상호 교환하는 데이터의 무결성을 목적으로 한다. TLS 는 핸드쉐이크를 통해 사용할 암호 알고리즘들을 합의하고 인증서를 통해 상대를 인증하며, 최종적으로 대칭키와 메시지 인증 코드(MAC) 키를 합의한다. 합의된 키를 통해 데이터는 암호화되고 무결성이 보장된다.

2. X.509 인증서

인증서는 신뢰할 수 있는 제 3 자를 통해 신원과 공개키를 연결한다. TLS 내에서 사용되는 것은 인증서의 표준 양식인 X.509 버전 3 (X.509v3)[5]이 두루 쓰인다.

인증서는 인증 대상을 서술한다. 기본적으로 인증 대상의 이름과 공개키, 인증 기관의 이름과 서명이 들어간다. 또한 하나의 인증서로 인증 대상의 여러 이름을 포괄하기 위해 Subject Alternative Name (SAN)을 정의하여 대상의 모든 이름을 나열한다.

수신자의 인증서 검증은 다음 세 가지의 과정을 거친다. 첫째, 인증서가 신뢰하는 대상이 발급한 것인지 확인한다(신뢰 체인 검증). 둘째, 인증서가 인증할 대상의 것인지 확인한다(이름 검증). 셋째, 인증서가 유효한지 확인한다(폐기 여부 검증). 이 세 가지가 충족되면 상대를 신뢰하게 된다.

3. TLS Certificate Compression

IETF 의 TLS WG 에서 현재 논의되고 있는 TLS-CC 는 TLS 핸드쉐이크에서 압축된 인증서를 활용하는 TLS 확장이다. 이 프로토콜은 지연 시간 단축과 성능 향상을 목적으로 한다. 서버-클라이언트 양단은 핸드쉐이크에서 TLS-CC 의 사용과 압축 알고리즘을 합의하고 압축된 인증서를 통해 인증을 수행한다.

Ⅲ. 인증서 현황

본 장에서는 scans.io 에서 2017 년 11 월 21 일에 수집한 인증서들의 크기 및 5 가지 알고리즘을 적용했을 때의 크기 변화를 분석한다. 인증서 인코딩 방식에는 PEM 과 DER 두 가지 방식이 있으며 이에 따라 크기가 달라진다. 본 분석에서는 네트워크에서 인증서가 전송될 때의 인코딩 방식인 DER 에 국한하여 살펴본다.

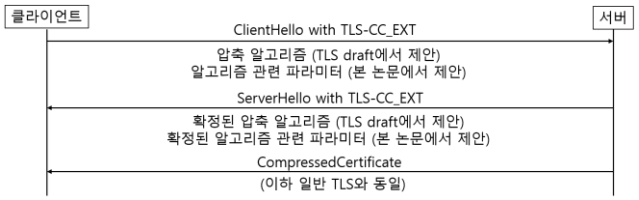
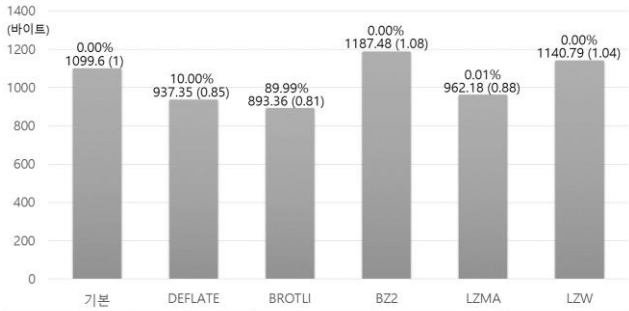


그림 1 TLS-CC 의 핸드쉐이크. 처음 클라이언트는 TLS-CC 의 지원 여부와 가능한 압축 알고리즘을 보내면 서버는 알고리즘을 확정하고, 해당 알고리즘으로 압축된 인증서를 보낸다.



인증서 수집 일자	인증서 개수	평균 인증서 크기	최대 크기	최소 크기
2017-11-21	291,742	1099.60	19,467	269

그래프 1 2017 년 11 월 21 일자로 수집된 인증서 291,742 개에 대한 압축 알고리즘 적용 결과. 막대 그래프 위의 비율은 전체 인증서에 대해 해당 알고리즘이 가장 많은 압축률을 보이는 비율(%)이며, 아래 값은 압축된 인증서의 평균 크기이고 괄호 안은 본래 인증서 크기 대비 압축된 인증서의 크기의 평균 비율.

1. 인증서 크기 분석

그래프 1 의 하단에는 인증서의 평균/최대/최소 크기를 정리하였다. 인증서의 크기는 평균 1K 바이트 정도이며 작게는 269 바이트에서 크게는 19K 바이트에 달한다. 크기가 큰 인증서들은 SAN 이 많기 때문이다. 이 인증서의 소유자는 여러 국가코드 최상위 도메인을 가진 대규모 회사이거나 여러 도메인을 대행하는 CDN 이다. 따라서 이 인증서들은 무시할 수 있는 이상치가 아니다.

이를 통해 인증서를 압축하는 것이 서버와 클라이언트 모두에게 유리할 것이라고 예상할 수 있다. 서버 입장에서는 인증서로 전송하는 트래픽이 대략 80%선이 되어 네트워크에서 사용하는 대역폭을 감소시킬 것으로 기대할 수 있다. 한편, 클라이언트는 수신하는 패킷 수가 감소하여 핸드쉐이크에 필요한 시간과 전원 소모량이 줄어들 것으로 기대할 수 있다.

2. 알고리즘에 따른 압축률 분석

다음으로 압축 알고리즘에 따른 인증서 압축률을 검토하였다. 압축 대상은 최신 인증서 데이터셋으로 하였다. 적용한 알고리즘은 총 5 개로 [3]에서 기본으로 정한 deflate, brotli 와 함께 bz2, lzma, lzw 을 활용하였다. 이 실험은 [3]에서 선정한 알고리즘의 적절성을 검토하고 추가할 만한 알고리즘과 선택 우선순위를 확인하기 위함이다.

그래프 1 의 막대 그래프는 위 실험의 결과이다. 결과를 보면 알 수 있듯이, brotli 를 적용했을 때 가장 좋은 압축률을 보이는 것을 알 수 있다. 그 다음으로 deflate 와 lzma 순으로 압축률이 좋았다. 이 결과는 적절한 알고리즘 선택 우선순위를 보여준다. 한편으로는, 압축 알고리즘으로써 bz2 나 lzw 는 사용하지 않는 것이 좋다는 것을 알 수 있다.

IV. 구현 및 실험

본 논문이 가정하는 시나리오는 저사양 기기가 클라이언트로서 TLS-CC 를 사용하여 고성능의 웹서버들과 암호 통신을 수행하는 경우이다. 이는 TLS-CC 가 저사양 기기에서도 가능한지 확인하기 위함이다.

구현 환경은 다음과 같다. 서버는 Intel Xeon CPU E3-1245 를 장착하고 32GB 메모리를 가지며, 저사양 기기는 TI CC3200 SoC 를 사용하였다. 이는 ARM Cortex M4 80MHz 의 MCU 를 사용하며 256KB 의 RAM 과 WiFi 모듈을 가진다. 암호화 라이브러리는 WolfSSL-3.12.0 을 확장하고, zlib 라이브러리와 결합하였다. 압축 알고리즘은 deflate 를 사용하였다.

저사양 기기의 경우, 한정된 메모리 내에 적용이 가능해야 한다. 현재 제안된 TLS-CC 는 단순히 압축 알고리즘만 협상하기 때문에 메모리가 작은 기기는 TLS-CC 를 도입하기 어렵다. 따라서 압축 윈도우 크기와 메모리 제한 등의 파라미터를 메시지에 추가하여 협의할 필요가 있다(그림 1 참조). 서버는 합의된 압축 알고리즘과 파라미터에 따라 인증서를 압축하여 전송하면 된다.

개체	압축 알고리즘	TLS 핸드쉐이크 바이트 수	비고
서버	기본	4,300	-
	DEFLATE	3,757	12.6% 감소

표 1 서버 수행에 따른 트래픽 이득

개체	압축 알고리즘	시간 (ms)	전원 소모량 (mJ)
클라이언트	기본	2,136.6	582.1
	DEFLATE	1,987.6 (6.97% 감소)	555.2 (4.62% 감소)

표 2 클라이언트 수행에 따른 시간/전력 이득

표 1 과 2 는 실험 결과를 보여준다. 서버는 인증서 압축으로 TLS 핸드쉐이크 과정에서 전송하는 트래픽이 12.6% 정도 감소한다. 클라이언트의 경우 6.92%의 시간 이득과 4.62%의 전력 이득을 본다. 그러나 클라이언트의 경우, 무선 상태에 따라 큰 변동이 발생하기 때문에, 위 이득이 눈에 띄지 않는 경우도 많이 있다. 따라서 TLS-CC 는 서버에게 이득이 분명하며, 클라이언트의 성능 상 이득은 미미하다. 하지만 파라미터 조율까지 가능하다면 TI CC3200 SoC 수준의 기기까지 포괄할 수 있다.

VI. 결론

본 논문에서는 인증서의 크기 현황을 분석하고 인증서 압축을 활용한 TLS-CC 를 시나리오에 맞게 적용하여 적절성과 보완점을 검토해 보았다. 인증서 압축으로 서버의 트래픽 부하는 12.6% 감소하였으며, 알고리즘의 파라미터 조율도 시행하면 저사양 기기도 포괄할 수 있다는 것을 확인하였다.

ACKNOWLEDGMENT

이 논문은 2017 년도 정부(교육부)의 재원으로 한국연구재단의 지원을 받아 수행된 기초연구사업임(2017R1A6A3A11032626)

참 고 문 헌

[1] Felt, Adrienne Porter, et al. "Measuring HTTPS Adoption on the Web.", USENIX Security 2017.

[2] D. Naylor et al. "The cost of the S in HTTPS." *Proceedings of the 10th ACM International on Conference on emerging Networking Experiments and Technologies*. ACM, 2014.

[3] A. Ghedini et al. “Transport Layer Security (TLS) Certificate Compression”, RFC draft, IETF, 2017

[4] Dierks, Tim, and Eric Rescorla. "Rfc 5246: The transport layer security (tls) protocol." *The Internet Engineering Task Force*(2008).

[5] Housley, Russell, et al. *Internet X. 509 public key infrastructure certificate and certificate revocation list (CRL) profile*. No. RFC 3280. 20