SNI 차단 및 암호화 연구

윤재웅* 김경호** 서화정**†
*한성대학교 컴퓨터공학부
**한성대학교 대학원 IT융합 공학부

요 약

- 불법 사이트 차단을 명분으로 진행중인 인터넷 검열이 표현의 자유를 억압할 수 있다.
- 올해 2월부터 HTTPS를 차단하기 위해 고도화된 전략으로 SNI차단을 도입했다.
- 기존 차단 방식들의 동향 및 현재 진행중인 차단 방식에 대해 알아본다.
- TLS 1.3에서 제안된 ESNI 암호화 방식에 관해 알아본다.
- 드래프트에서 Open Issue로 언급된 ESNI를 통한 0-RTT의 가능성에 대해서 확인해본다.

차단

• DNS 차단

- 클라이언트의 요청을 파밍(Pharming)기법을 이용하여 ‘Warning’ 사이트로 리다이렉션시킨다.
- DNS 차단은 해외 서버를 이용한 우회가 가능하다.

• HTTP 차단

- 암호화 되지 않은 평문으로 통신한다.
- 해외 DNS로부터 IP 주소를 수신해도 ISP(Internet Server Provider)에게 해당 주소를 요청시 차단할 수 있다.
- TLS(Transport Layer Security)로 암호화한 HTTPS를 통한 우회가 가능하다.

• HTTPS(SNI) 차단

- SNI는 TLS의 확장 표준 중 하나로써 인증서에서 허용하는 방식이다.
- 웹 서버에서 다수의 도메인의 웹 사이트를 서비스하는 경우에도 인증서를 사용한 HTTPS 사용을 가능하게 한다.
- HTTP를 암호화 하기 전 ClientHello 패킷에서 평문 노출된다.
- TLS 1.3에서 SNI 암호화인 ESNI(Encrypted SNI) 제안(드래프트)이 되었다.

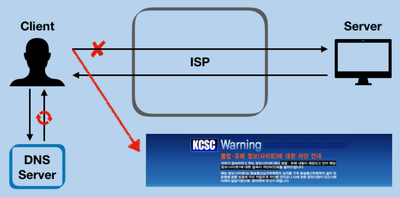

fig 1. DNS 차단 방식

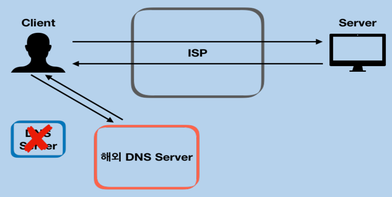

fig 2. 해외DNS를 통한 차단 우회

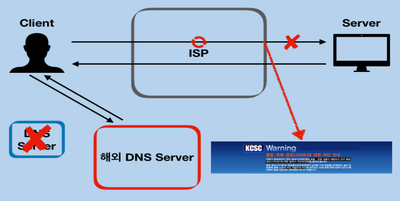

fig 3. HTTP 차단 방식

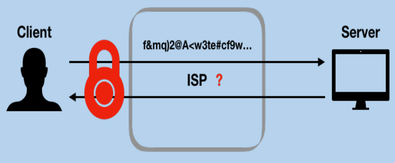

fig 4. HTTPS를 통한 차단 우회

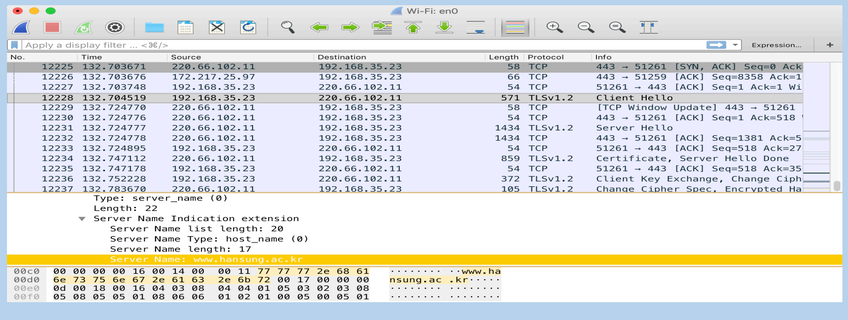

fig 6. SNI의 평문 노출

ESNI

1. 암호화를 하기 위해서 웹 서버는 신뢰할 수 있는 DNS 공개키를 미리 올려놓는다.
2. 클라이언트가 DNS에 요청 시 TXT 레코드를 통해 ESNIConfig를 받는다.
   (버전, 키 갱신, 공개키, 고려할 수 있는 확장, 서버 등의 정보)
3. 웹 서버의 공개키와 클라이언트의 개인키로부터 대칭키를 유도한다. *DH(Diffie–Hellman key exchange)*
4. 유도한 대칭키를 이용해 ClientEncryptedSNI를 암호화 한 후 클라이언트의 ClientHello 패킷을 보낸다.
   (키, 암호 해시, ESNI 등의 정보)
5. 웹 서버는 자신의 개인키와 클라이언트의 공개키로부터 동일한 대칭키를 유도하여 복호화 한다. *DH(Diffie–Hellman key exchange)*

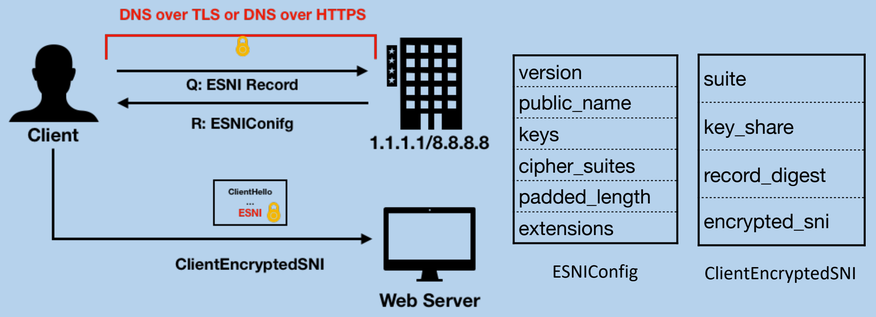

fig 7. ESNI 암호화 과정

• DNS 암호화 - *ESNI를 이용하면서 평문으로 통신하는 DNS를 사용하면 SNI를 암호화하는 의미가 없다.*

- DNSSEC : DNS 데이터 대상의 “데이터 위조-변조 공격” 방지를 하기 위한 인터넷 표준기술이다. (전자서명)
- DNS-over-TLS : 클라이언트와 DNS 서버 간의 TLS 프로토콜 기반 암호화된 통신을 통해 암호화한다. (port : 853)
- DNS-over-HTTPS : DNS 요청을 평문이 아닌 HTTPS를 통해 접근하는 방법으로 DNS 요청을 HTTP 요청으로 위장하여 높은 보안성 효과 및 확장성을 제공한다. (port 443)

0-RTT

• TLS 1.2 재연결 - *HTTP를 암호화하는 과정을 매번 Full Handshake 하는 것은 비효율적이다.*

- Session ID : 서버가 발행한 값들을 모두 메모리에 기록한다.
- Session Ticket : 클라이언트가 해당 내용을 기억하고 서버로 전달한다.

* Session ID, Session Ticket 모두 초기 Full Handshake 과정에서 서버로부터 값을 할당 받는다.
* 클라이언트와 서버가 모두 Session Ticket을 지원하지 않다면 Session ID를 사용하게 된다.(Session Ticket 권장)

fig 8. Full Handshake

fig 9. Session ID, Session Ticket 활용

• TLS 1.3 0-RTT

- 시범적 도입으로 기존 방법보다 빠르게 처리할 수 있는 가능성을 제시했다.

0-RTT는 기존의 Session ID, Session Ticket과 같은 역할을 수행하는 PSK(Pre-Shared Key)를 클라이언트의 ClientHello 단계에서 Application Data를 미리 암호화 해서 일방향적으로 보내는 방식이다.

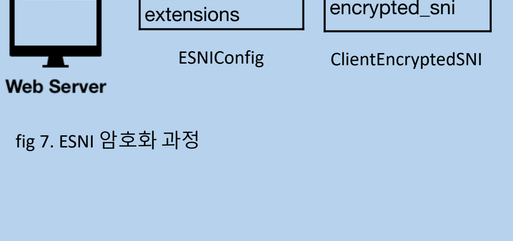

fig 9. TLS 1.3 0-RTT

• ESNI 0-RTT

- ESNI 드래프트에서 Open Issue로 언급되었다.

SNI를 암호화하기 위해 웹 서버의 공개키를 사전에 신뢰할 수 있는 DNS 서버에 올리는 과정을 통해 클라이언트는 웹 서버와 통신하기 전 이미 자신과 웹서버의 공개키를 *DH(Diffie–Hellman key exchange)* 방식을 통해 ESNI-PSK를 유도할 수 있다.
이는 기존의 TLS 1.3의 0-RTT의 PSK를 DNS 프로토콜을 통해 Full Handshake 과정없이 [fig 7]과 같이 ESNI 암호화와 같이 Application Data를 암호화 하여 보낼 수 있다.