논문 15-41-01-15 The Journal of Korean Institute of Communications and Information Sciences '16-01 Vol.41 No.01
http://dx.doi.org/10.7840/kics.2015.41.1.101

쿠키 기반의 TLS/SSL 인증서 공개키의 확인

박 준 철◆

Cookie-Based Identification of the Public Keys of TLS/SSL Certificates

Jun-Cheol Park◆

요 약

HTTP 쿠키(cookie)를 활용하여, 웹 사이트의 인증서 검증이 실패하였을 때 해당 사이트의 공개키를 확인하는 방법을 제시한다. 제안 방식은 속임수에 의해 가짜 사이트로 접속하도록 유도하는 피싱(phishing) 공격에 대해 효과적인 방어 수단이 된다. 제안 방식은 간단한 사용자 인증정보의 입력 처리와 쿠키의 암호화 및 검증 과정의 구현을 요구하나, 정상 동작 상황에서 브라우저 및 웹 사이트 운용 서버 어느 쪽에도 성능상의 추가 부담을 거의 지우지 않는다.

Key Words : TLS/SSL, cookie, public key, authentication, phishing

ABSTRACT

We propose a HTTP cookie-based identification of the public keys of Web sites for the case of failure to validate certificates. The proposed scheme effectively protects users from the phishing attacks of inducing them to access bogus sites. It incurs little performance overhead on the browser and the server of Web sites. It requires to implement the input processing of user credentials and the encryption and verification of cookie values, though.

Ⅰ. 서 론

HTTPS를 통하여 접속한 사이트의 X.509 인증서(사이트의 공개키 및 CA(Certificate Authority)의 서명 포함)를 검증하는데 실패 했을 때(유효기간 만료, 스스로가 서명한 인증서, 다른 사이트의 인증서, 신뢰할 수 없는 CA가 서명한 인증서, 개인키 유출된 인증서, 기타 이유로 폐기된 인증서 등의 이유로) 브라우저는 사용자에게 경고[1] 창을 띄우고 사용자가 위험을 감수하고라도 계속 진행할 것인지 묻는다. 선택의 기로에서, 대다수 일반 사용자들은 자신의 선택이 어떤 최악의 결과를 가져올지 큰 고민 없이 계속 진행함을 선택하는 것이 일반적이다.

기존 TLS/SSL[2] 인증서 검증 관련 연구는 인증서 검증 과정의 오류 및 브라우저 간 차이를 확인하기 위해 인증서 테스트 케이스를 자동 생성하는 연구[3], 공개된 로그를 통해 인증서의 위조를 어렵게 하는 시스템 구조 연구[4] 등을 들 수 있으나, 인증서의 핵심 정보인 사이트 공개키를 별도의 수단을 통해 확인 가능하도록 하려는 시도는 없었다. 본 논문은 HTTP 쿠키를 활용한 웹 사이트 공개키의 추가 확인 방식을 제안한다. 이 방식은 인증서 공개키가 상대방 사이트 것이 아님이 밝혀질 때, 사용자에게 접속 중단을 강력히 요청하여 피싱[5] 공격의 방어에 유용하게 사용된다. 제안 방식은 인증서에 대한 검증이 성공할 경우에는 실행되지 않기 때문에 정상 상황에서의 성능 부담은 없으며, 추가 확인 과정 자체도 수 회 이내의 대칭키 암/복호화 및 해쉬 함수 적용에 불과하므로 성능상의 부담이 적다.

Ⅱ. HTTP 및 쿠키 동작 개요

HTTP 프로토콜은 상태를 저장하지 않는데, 쿠키(cookie)를 이용하면 웹 서버가 자신에 접속하는 브라우저 관련 정보를 저장하는 것이 가능해진다. 쿠키는 브라우저를 통해 클라이언트 컴퓨터에 저장되는 소규모의 텍스트 파일이다. 쿠키에는 데이터인(cookie-name, cookie-value) 쌍이 필수로, 만료시점(expiry date), 쿠키가 보내질 서버의 도메인 및 경로(path) 정보 등의 부가 정보가 옵션으로 포함된다.

서버는 쿠키를 생성하고, 이를 브라우저에 전달하기 위해서 아래와 같은 Set-Cookie 헤더 필드를 포함하는 HTTP 응답(response)을 보낸다.

```
Set-Cookie: SID=31d4d96e407aa42;
            Path=/; Domain=example.com
```

※ 본 논문은 2014학년도 홍익대학교 학술연구진흥비에 의하여 지원되었음.
◆ First Author : Department of Computer Engineering, Hongik University, jcpark@hongik.ac.kr , 종신회원
논문번호 : KICS2015-10-337, Received October 16, 2015; Revised November 17, 2015; Accepted January 15, 2016

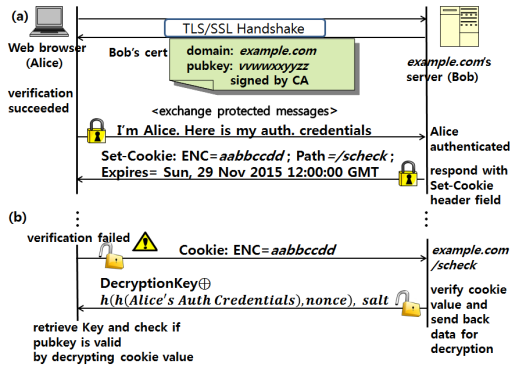

그림 1. (a) HTTPS에서의 Alice 인증 후 쿠키 설정, (b) 쿠키 기반의 사이트 공개키 확인
Fig. 1. (a) HTTPS Cookie Set after Alice's being Authenticated, (b) Cookie-based Identification of example.com's public key

이제 브라우저는 이후의 HTTP 요청(request)을 위의 도메인 및 경로에 해당하는 서버 사이트로 보낼 때, Cookie: SID=31d4d96e407aa42와 같이 부가적 정보를 제외하고 쿠키의 네임 및 값의 쌍을 헤더 필드에 포함시켜서 전송하게 된다. 쿠키의 Expires 옵션은 쿠키 값의 유효 기간을 규정하며, Domain 옵션은 쿠키를 포함시켜 보낼 서버의 도메인을 지정하고, Path 옵션은 쿠키가 서버의 지정된 도메인 경로와 매치되는 사이트로만 전송되도록 한다. Secure 옵션은 HTTPS 프로토콜로 요청을 보낼 때에만 쿠키가 전달되도록 한다.

만료 기간이 설정된 쿠키는 사용자가 강제로 삭제하지 않는 한 세션이 종료되더라도 만료 기간이 될 때까지 지워지지 않는다.

Ⅲ. 웹 사이트 공개키의 추가 확인 기법

제안 방식의 기본 아이디어는 인증이 완료된 클라이언트의 정보를 웹 사이트의 공개키와 결합한 후 대칭키 암호화시킨 값을 쿠키로 지정하여, 추후 해당 웹 사이트의 공개키에 대한 추가 확인이 필요할 때 이 쿠키 값을 활용하는 것이다. 이를 위해 서버는 사용자 인증 후에, $E(h(PK_S), ipaddr, h(credentials), time, K_C)$을 계산한다. 여기서, PK_S는 웹 사이트의 공개키, $ipaddr$는 클라이언트의 IP 주소, $credentials$는 클라이언트 사용자의 인증 정보로서 서버에 저장되는 형태(예: $h(pwd, salt)$), $time$은 클라이언트가 이 웹 사이트에 접속한 일시를 의미한다. 또한 $h(K_{SM}, ipaddr, nonce_S)$로 계산한 키 K_C는 대칭키 암호화(예: AES-256)를 위해 사용한다. 여기서 K_{SM}은 서버만이 알고 있는 마스터키 값이며, $ipaddr$는 클라이언트의 IP 주소, $nonce_S$는 K_C 값을 매번 갱신하기 위해 서버가 사용하는 일회용 nonce 값이다. 서버는 다음과 같이 Set-Cookie 헤더 필드를 지정하여 브라우저에게 전송한다(만료 기간은 현재부터 2~3주 후).

Set-Cookie:ENC= $\underline{E(h(PK_S), ipaddr, h(credentials), time, K_C) \| nonce_S}$; Path=/scheck;
Expires=Sun, 29 Nov 2015 12:00:00 GMT

단, 밑줄 친 부분은 Base64로 인코딩 한 후 전송한다(이하 밑줄은 해당 비트열을 Base64로 인코딩한 결과를 나타냄). 쿠키 ENC의 값(value)은 클라이언트 입장에서 의미 없는 랜덤 값으로 보이며, 서버 입장에서는 암호화한 후 기억할 필요가 없는 값이 된다. Path 옵션을 /scheck로 지정함으로써 이 쿠키는 서버의 추가 확인 전용 페이지가 위치한 도메인의 /scheck 위치로만 보내지게 되며, Expires 옵션은 시점을 2~3 주 이후로 지정하여 그 사이의 쿠키 자동 삭제를 막는다(그림 1).

3.1 서버에서의 쿠키 값 처리

서버의 /scheck 페이지에서는 전달된 쿠키 값 $\underline{E(h(PK_S), ipaddr, h(credentials), time, K_C) \| nonce_S}$을 다음과 같이 처리한다.

(1) 밑줄 친 부분을 다시 비트열로 바꾼 후, 고정된 크기의 $nonce_S$를 추출한다. 서버의 마스터키 K_{SM} 및 Cookie 필드를 전송한 클라이언트의 IP 주소 $IPaddr$과 $nonce_S$를 사용하여, $K = h(K_{SM}, IPaddr, nonce_S)$를 계산한다.

(2) K로 $E(h(PK_S), ipaddr, h(credentials), time, K_C)$의 복호화를 시도한다. 만약 복호화의 결과 $\overline{h(PK_S)}, \overline{ipaddr}, \overline{h(credentials)}, \overline{time}$ 를 얻었다면, $IPaddr = \overline{ipaddr}$ 이고, 해당 웹 사이트의 공개키를 해쉬한 결과가 $\overline{h(PK_S)}$와 같으며, $\overline{time}$ 이 과거의 시간인지를 확인한다. 이 검증은 자신(서버)이 수신된 쿠키 값을 이전에 암호화했으며, 이 값이 전송 도중에 변조되지 않았음을 확인하는 역할을 한다. 이때 서버는 K_{SM} 만을 기억하고 있으면 복호화에 필요한 키를 즉각 만들어낼 수 있다.

(3) (2)의 확인이 성공적으로 끝나면, 서버는

$h(\overline{h(credentials)}, nonce_S) \oplus K_C, salt$(서버에서 $salt$를 써서 인증정보를 저장한 경우, 아니면 $salt$ 제외)를 HTTP 응답 형태로 브라우저에게 전송한다.

3.2 브라우저에서의 쿠키 값 처리

수신한 $h(\overline{h(credentials)}, nonce_S) \oplus K_C, salt$를 이용하여 브라우저는 서버(사용자의 인증정보 및 최근 접속정보를 알고 있다고 확인된)가 제공한 쿠키 값에 포함된 사이트 공개키의 해쉬 값을 추출한다. 이 해쉬 값을 통하여 인증서에 포함된 공개키가 사이트의 공개키가 맞는지 확인한다.

(1) 클라이언트 사용자의 인증정보(어떤 인증정보를 사용할지 서버와 미리 약속된) X를 입력받는다.

(2) 수신한 $salt$와 X를 이용하여, $h(X,salt)$(만약 $salt$가 없으면 $h(X)$)를 계산하고, 저장된 쿠키 값의 $nonce_S$를 이용, $h(h(X, salt), nonce_S)$를 계산한다.

(3) 수신한 $h(\overline{h(credentials)}, nonce_S) \oplus K_C$로부터 $h(\overline{h(credentials)}, nonce_S) \oplus K_C \oplus h(h(X, salt), nonce_S) = K^*$ 를 얻는다. 사용자의 인증정보가 서버의 것과 일치한다면 $K^* = K_C$ 이 될 것이다. 저장된 쿠키 값의 $E(h(PK_S), ipaddr, h(credentials), time, K_C)$ 부분을 K^*로 복호화 한다. 복호화의 결과로, $\widehat{h(PK_S)}, \widehat{ipaddr}, \widehat{h(credentials)}, \widehat{time}$를 얻었다면, 자신의 IP 주소가 $\widehat{ipaddr}$, $h(X,salt) = \widehat{h(credentials)}$이며, 자신이 상대방 웹 사이트를 정상적으로 방문한 마지막 시각이 $\widehat{time}$ 임을 확인한다.

(4) (3)의 확인이 성공적으로 끝나면, 수신한 인증서에 포함된 공개키 Z로 $h(Z) = \widehat{h(PK_S)}$ 인지를 확인한다. 일치할 때, Z가 상대방 웹 사이트의 공개키라고 최종 판정한다.

Ⅳ. 평가 및 결론

제안 방식은 접속한 웹 사이트의 인증서 검증이 실패한 경우에, 이전에 해당 사이트에 접속하여 인증을 받은 기록 정보를 활용하여 접속 사이트의 공개키 정보를 확인할 수 있도록 한다. 피싱 공격 등에 의해 유도된 유해 사이트는 브라우저가 보내는 암호화된 쿠키 값을 복호화할 수 없기 때문에 브라우저에게 적절한 값을 반환할 수 없다. 유해 사이트가 브라우저에 의해 복호화 검증이 실패할 값을 반환한다면, 브라우저는 사용자에게 인증서 공개키의 추가 확인이 실패로 끝났음을 알리고 접속 중단을 강력히 권장할 수 있다. 만약 유해 사이트가 중간자 역할을 하며 정상 사이트와 브라우저 간의 통신 내용을 중계만 한다면, 유해 사이트는 (1) 자신을 정상 사이트로 믿도록 브라우저를 속일 수 없으며, (2) 두 번의 해쉬 함수 및 대칭키 암호화로 보호된 사용자의 인증정보를 추출할 수도 없다. 만약 유해 사이트가 중간자 역할을 하며 암호화 쿠키 및 HTTP 응답 내용을 변조 후 재생한다면 (1) K_{SM} 및 K_C를 모르기 때문에 정상 사이트 서버를 속이는 변조를 가할 수 없으며, (2) 사용자 인증정보를 모르기 때문에 브라우저를 속이는 HTTP 응답을 조작할 수도 없다. 물론 중간자가 내용을 무작위로 바꾸어 제안 방식 검증이 실패하도록 만들 수는 있으나, 이는 유해 사이트로의 접속을 유인하려는 공격 목적에 반하므로 공격자에게 이익이 되지 못한다.

제안 방식의 적용을 위해 간단한 사용자 인증정보의 입력 처리와 쿠키의 암호화 및 검증 과정의 구현이 필요하나, 쿠키 이외의 별도의 데이터 생성, 유지는 필요치 않다. 웹 서버 측은 사용자 인증이 완료된 상황에서 하나의 쿠키만을 대칭키 암호화를 통해 생성하고, 암호화에 사용되는 매번 바뀌는 키들은 전혀 기억할 필요가 없다. 따라서 제안 방식이 적용되기 전의 정상 동작 상황에서 서버나 브라우저 측의 성능상의 추가 부담은 거의 없다.

References

[1] D. Akhawe and A. Felt, “Alice in warningland: a large scale field study of browser security warning effectiveness,” in *Proc. USENIX Security*, pp. 257-272, 2013.

[2] C. Meyer and J. Schwenk, “SoK: lessons learned from SSL/TLS attacks,” in *Proc. WISA*, pp. 189-209, Aug. 2014.

[3] Y. Chen and Z. Su, “Guided differential testing of certificate validation in SSL/TLS implementations,” in *Proc. ESEC/FSE 2015*, pp. 793-804, Aug.-Sept. 2015.

[4] P. Szalachowski, S. Matsumoto, and A. Perrig, “PoliCert: secure and flexible TLS certificate management,” in *Proc. ACM CCS 2014*, pp. 406-417, Nov. 2014.

[5] S. Kim, J. Kang, and Y. Kim, “Countermeasures against phishing/pharming via portal site for general users,” *J. KICS*, vol. 40, no. 6, pp. 1107-1113, Jun. 2015.