SSL MITM 프록시 공격에 대한 효과적 방어방법

(An Effective Protection Mechanism for SSL Man-in-the-Middle Proxy Attacks)

임 차 성 † 이 우 기 ††
(Chasung Lim) (Wookey Lee)

조 태 창 †††
(Tae-Chang Jo)

요 약 신용카드 정보나 공인 인증서들이 웹을 통해 전송되어지는 현재 전자상거래 시스템에서 사용자의 의도와 다르게 클라이언트가 웹 프록시 서버를 경유하게 되거나 프록시 서버의 경로가 변경될 경우가 종종 발생한다. 이때 전송되는 기밀 정보가 SSL(Secure Sockets Layer) 또는 TLS(Transport Layer Security)프로토콜을 통해 암호화 되어 전송 되어도 프록시 서버에서 인증서 변조를 통하여 계좌번호나 신용카드 비밀번호를 스니핑 당하는 위험에 노출된다. 본 논문에서는 현재 사용되고 프록시 정보 변조 해킹에 의해 무방비 상태의 신용카드 결제 보안 시스템에 대해서 분석하고 이를 방지하기 위한 인증 프록시 서버와 프록시 변조 MITM(Man-In-The-Middle) 공격 방지 방법에 대해서 제안한다.

키워드 : SSL, TLS, MITM, Proxy, HTTPS

· 본 연구는 (주)소만사의 개인정보 보호를 위한 보유 통제 시스템 사업의 연구 결과물임을 밝힙니다.
· 이 논문은 제36회 추계학술발표회에서 'SSL Man-in-the-Middle Proxy 공격에 대한 효과적 방어방법'의 제목으로 발표된 논문을 확장한 것임

† 학생회원 : 인하대학교 산업공학과
steady79@gmail.com
†† 종신회원 : 인하대학교 산업공학과 교수
trinity@inha.ac.kr
(Corresponding author임)
††† 정 회 원 : 인하대학교 수학과 교수
taechang@inha.ac.kr

논문접수 : 2010년 1월 20일
심사완료 : 2010년 3월 28일

정보과학회논문지: 컴퓨팅의 실제 및 레터 제16권 제6호(2010.6)

Abstract In current e-commerce system, it happens that client's confidential information such as credit card numbers, pin numbers, or digital certificate may pass through a web proxy server or an altered proxy server without client's awareness. Even though the confidential information is encrypted and sent through SSL(Secure Sockets Layer) or TLS(Transport Layer Security) protocol, it can be exposed to the risk of sniffing by the digital certificate forgery at the proxy server, which is called the SSL MITM(Man-In-The-Middle) Proxy attack. In this paper, current credit card web-payment systems, which is weak at proxy information alternation attack, are analyzed. A resolution with certificate proxy server is also proposed to prevent the MITM attack.

Key words : SSL, TLS, MITM, Proxy, HTTPS

1. 서 론

오늘날 정보 통신 기술의 발달은 사회 전반에 놀라운 변화를 주도하는 동인이 되고 있으며, 그중에서도 기존에는 사무실에서 처리해야만 되던 업무를 개방된 인터넷을 통해 빠른 속도로 처리할 수 있게 되었다. 하지만 인터넷의 편의성에 못지않게 사이버테러, 웜(Worm)바이러스, DDos(Distributed Denial Of service)공격, 크래킹(Cracking) 등의 피해가 속출하고 있으며, 고객의 개인정보를 해킹당한 옥션 사건과 같이 크래커에 의한 개인정보 유출이라는 문제가 점점 증가하고 있다.

오늘날 대부분의 전자상거래 사이트에서는 SSL(Secure Sockets Layer)와 TLS(Transport Layer Security) [1-3] 프로토콜로 서버와 클라이언트 사이의 공유채널을 암호화하여 데이터를 전송하지만, 이는 완벽한 보안 시스템이라고 볼 수 없다. 프록시 서버(Proxy Server)는 캐싱을 통한 로드 분산 등의 효용성 때문에 많이 활용되지만, 악성 프로그램이나 스크립트에 의해 MITM[4,5]을 하기 위한 도구로 악용될 수도 있다. MITM이란 클라이언트와 서버 사이에서 스니핑 하는 행위를 말하는 것으로, 암호화된 상태로 전송되는 기밀정보를 인증서 변조 방법을 이용하여 스니핑할 수 있는 방법 등이 포함된다.

본 논문에서는 웹브라우저 등을 사용하여 웹으로 전송되는 기밀정보가 웹 프록시를 걸쳐감으로 인해 발생할 수 있는 위험성에 대하여 분석하고 그러한 위험에 관한 새로운 방어 방법을 제안하고자 한다.

2. 웹 프록시 서버

프록시 서버는 다음과 같은 기능을 하기 위하여 사용된다. 우선 사용자의 접속정보를 프록시 캐시(Proxy Cash)에 일시 보관하고 여러 사용자가 이를 공유하여 망의 부하와 웹서버의 부하를 감소시키는 역할을 하며,

동시에 사용자에 대한 서비스 속도를 개선하는 기능을 제공할 뿐만 아니라, 제한적인 대역폭을 갖는 구간에서 인증서버 역할을 대신하여 인증을 해주는 역할을 하고 있다. 프록시 서버는 웹 캐싱 방식을 주로 사용하는데, 웹 캐싱이란 HTML문서와 같은 웹 객체의 일시적인 저장으로 사용자가 요청한 객체가 캐시에 있는 경우, 해당 서버에 접속하지 않고 캐시에 저장된 자료를 사용자에게 전송하는 기능이다. 이 기능의 장점으로는 대역폭 소비(bandwidth consumption)절감, 서버 부하 감소, 그리고 지연시간의 감소라는 세 가지가 장점이 있는 반면, 악용될 경우에는 MITM 공격의 도구[6]로 사용될 수 있다. 클라언트들이 프록시 서버를 경유하여 서비스 받는 예로는 클라이언트-프록시 서버-웹서버 통신형태 네트워크 환경은 LAN의 대역폭이 WAN의 대역폭보다 큰 형태로 공공기관이나 기업 등에서 많이 사용되고 있으며 이 경우 프록시 서버는 방화벽기능[7]을 같이 하기도 한다.

HTTP-Cache-Poisoning을 하기 위한 웹 캐싱의 경우는 크게 다음 두 가지의 경우로 나누어 설명할 수 있다.

첫째, 캐싱은 클라이언트 어플리케이션에서 실행 될 수 있다. 웹 브라우저에 가상적으로 구현되는 캐시가 여기에 해당된다. 이것은 웹 브라우징의 속도를 높일 수 있지만 단지 한 사용자만을 위한 것으로 효율성은 높지 않다.

둘째, 클라이언트와 서버의 사이에 프록시 서버를 두는 것이다. 대부분의 프록시 캐시가 이와 같은 형태를 이루며 프록시 캐시는 여러 클라이언트에게 서비스를 제공함으로써 클라이언트의 처리속도를 빠르게 해주는 역할을 한다. 프록시 캐시가 웹 문서를 요청 받았을 때 메모리, 디스크 혹은 다른 곳의 문서로 응답이 가능하다면 실제 서버로의 접속이 없이 클라이언트에게 응답한다. 만약 캐시에서의 응답이 불가하다면 프록시 서버는 직접 웹 서버로 접속해서 문서를 가지고 와서 캐시에 저장한다. 프록시 캐시에 저장 공간이 실제 저장 가능한 양을 초과한다면 프록시 캐시는 문서를 대체해야만 한다. 일반적으로 캐시 교체정책은 요청되는 모든 문서 중에서 캐시에 의해 성공적으로 서비스되는 문서의 비율을 최대화하려고 하는 알고리즘을 사용한다. 이러한 프록시 서버의 유용성 때문에 많은 조직에서 프록시 서버를 사용하지만 악의적인 크래커에 의해 프록시서버를 악용될 경우에는 심각한 보안상의 허점이 생기게 된다.

3. HTTP-Cache-Poisoning 취약성 분석

일반적으로 클라이언트는 관리에 의해 임의로 프록시 설정이 되어 있지 않다면 웹서버를 통해 직접 서비스를 받게 된다. 하지만 악성 스크립트에 의해서 프록시 설정이 변경될 경우 프록시 서버로 HTTP request 스니핑 및 변조가 가능하고 서버에서 보안을 위해 SSL(Secure Socket Layer) 통신을 한다하여도 인증서 변조[8,9]로 기밀정보의 내용이 스니핑 될 수 있다.

3.1 레지스트리 변조로 인한 프록시 설정 변경

본 연구에서는 클라이언트의 레지스트리 값을 변경하면 설정된 값으로 클라이언트의 프록시 설정 레지스트리 값이 변경되고 클라이언트는 프록시 서버를 경유 하여 웹브라우저로 인터넷을 하게 된다. 이때 프록시 서버에서는 웹브라우저를 통해 전송되는 값들을 저장한다. 클라이언트는 자신이 인터넷을 할 때 프록시 서버를 경유하는지 알지 못한 채 인터넷을 사용할 수 있으며, 이때 프록시 서버에 전송되어 저장된 값은 변조가 가능하다. 이러한 방식으로 클라이언트가 악성코드에 의해 프록시 설정이 바뀌어서 프록시 서버에게 기밀정보를 전송하는 경우는 다음 세 가지로 세분할 수 있다.

3.1.1 스크립트에 의한 프록시 설정 변경

우선 스크립트에 의해서 레지스트리 값을 변경하는 방법이 있다. 해킹으로 인해 변조된 게시판이나 악성코드가 포함된 이메일에 스크립트 실행 파일을 실행하는 HTML 코드를 삽입한 글을 작성하는 방법으로 클라이언트 레지스트리 값이 변경될 수 있다. 통상적인 게시판이나 이메일은 보안상 기본적으로 스크립트를 작성할 수 없게 한 경우에도 HTML 코드로 스크립트 실행 파일을 실행하게 함으로서 클라이언트에서 스크립트를 실행하게 할 수 있다. 클라이언트는 악성프로그램이 작성한 게시판의 글이나 이메일을 열람 할 경우 HTML 코드가 실행되고, 원격 컴퓨터에 있는 스크립트 실행파일을 통해 클라이언트의 레지스트리 값을 변경할 수 있으며, 이 경우 클라이언트는 악성 프로그램이 설정한 프록시 서버를 통해서 웹서비스를 받게 된다.

3.1.2 ARP spoofing을 통한 프록시 설정 변경

ARP spoofing이란 게이트웨이의 MAC주소를 해커의 MAC주소로 바뀌었다고 전송해 줌으로서 클라이언트는 해커의 컴퓨터의 MAC 주소가 게이트웨이 MAC주소로 알고 패킷을 전송하는 것을 말한다. 이때 ARP spoofing을 당한 클라이언트가 전송하는 패킷은 클라이언트는 인지하지 못한 상태로 해커의 컴퓨터에 전송된다. 그러면 클라이언트가 원하는 웹서버에 접속하려고 할 때 해커가 설정해 놓은 웹 서버로 접속하게 해서 해커가 작성한 HTML 코드 속의 스크립트 실행 파일을 읽어오고, 해커의 컴퓨터에서는 스크립트 실행파일을 실행하여 클라이언트의 레지스트리 값을 변경할 수 있다. 이 경우 클라이언트는 웹에서 요청하는 모든 정보를 해커의 프록시 서버를 통해 웹서비스를 받게 된다.

3.1.3 악성 프로그램에 의해서 프록시 설정 변경

윈도우 계열 운영체제의 경우 레지스트리 값을 변경해서 프록시 설정을 가용 상태로 변경하게 된다. 레지스트리에서 프록시 서버 주소 값을 원격의 프록시 서버주소와 포트번호로 변경시킨다. 레지스트리 변경 예는 그림 2와 같다.

```
Language C#
My Computer\HKEY_CURRENT_UXER\Software\Microsoft\Windows
\CurrentVersion\Internet Settings
-> ProxyEnable                    REG_DWORD
0x00000001//(522)
-> ProxyServer                    REG_SZ
220.149.XXX.XXX:8080 //(523)
```

그림 2 레지스트리 변조 코드

프록시 변경 코드에 감염된 클라이언트에서 발생되는 절차는 다음과 같다. 클라이언트가 악성 코드나 스크립트에 의해 레지스트리 값이 변경되면 프록시 서버를 거쳐서 통신을 하게 된다. 이는 윈도우에 있는 기본적인 프록시 설정 기능이므로 보안 솔루션이나 바이러스 탐지 툴은 클라이언트의 프록시 설정이 악의적으로 변경되었는지 사용자에 의해 임의로 변경되었는지 판단하지 못하기 때문에 현재로서는 이를 탐지하는 기능이 없다.

클라이언트에 프록시 설정이 되면 인터넷 이용 시 웹 브라우저가 특정 포트를 열고 프록시 서버를 거쳐 웹 서버로부터 서비스를 받게 된다. 이 때 악의적인 해커에 의해 설정된 프록시 서버는 일반적으로 사용되는 프록시 서버의 기능을 하는 것이 아니라 클라이언트이 요청한 값을 받아서 캐쉬에 저장하고 웹서버에게 릴레이 하는 역할만 하게 되는데, 이때 사용자가 전송 값을 저장함으로서 해킹이 가능해진다. 또한 프록시 서버에서 클라이언트가 요청한 값을 변조하여 전송이 가능하기 때문에 웹사이트 요청 시 방향전환(redirection)이 가능하다. 방향전환이란, 예를 들어 클라이언트에서 bank.com으로 웹서비스를 요청 했을 시 프록시서버에서 bank.com이 아닌 www.hacker.com으로 해커가 원하는 사이트를 방문하게 함으로써 은행으로 착각하고 들어온 희생자는 개인정보를 해커의 웹사이트에 남길 수 있게 만드는 것이다. 이 때 클라이언트는 중간에서 프록시 서버를 거쳐서 가는지 눈치 챌 수 없기 때문에 해킹 당한다는 사실을 인지 할 수 없다. 따라서 프록시 서버를 이용하여 파라미터 변조, 쿠키값 변조, 인증서 변조, 클라이언트의 웹사이트 방문 캐쉬를 저장하는 등의 방법을 통해 개인정보를 스니핑 할 수 있다.

3.2 SSL통신을 하는 신용카드 정보 스니핑

프록시 서버를 설정하는 레지스트리 값이 악성코드에 의해 변조된 클라이언트는 해커의 프록시 서버를 통해서 웹 서비스를 받기 때문에 신용카드 결제 서비스에 접속 후 신용카드 결제 시 신용카드 정보, CVC값, 안심클릭 비밀번호 등 개인정보를 해커의 프록시 서버에게 전송한다. 해커의 프록시 서버는 캐시에 이를 저장하고 신용카드 회사에 클라이언트의 개인 정보를 전송한다.

신용카드 결제 시 상용 보안 솔루션이 실행되지만 이러한 해킹 방법은 윈도우에서 기본으로 제공하는 프록시 서버 기능의 설정을 변경한 방법이고 프록시 설정 자체는 해킹 프로그램이 아니므로 현행 대부분의 보안 솔루션에서는 이를 탐지할 수 없다.

3.3 프록시 서버를 이용한 SSL통신 인증서 변조

해킹 프로그램에 의해 감염된 클라이언트 즉 프록시 서비스를 받고 있는 클라이언트가 SSL 통신을 하는 웹 서버에 요청을 하게 되면 웹서버는 해커의 프록시 서버에게 자신의 공개키를 전송해 주고 해커의 프록시 서버는 자신이 만들어 놓은 변조된 공개키를 클라이언트에게 전송한다. 변조된 공개키를 수신한 클라이언트에서는 공개키가 신뢰된 곳에서 인증된 공개키가 아니라는 경고 메시지를 보낼 수 있으나 일반 사용자들은 공개키의 개념을 잘 알지 못하기 때문에 확인 버튼을 누르고 변조된 공개키로 웹서버로 전송되는 값을 암호화한 후 아이디 패스워드 등을 전송하는 경우가 많다. 이 때 해커의 프록시 서버는 변조된 공개키로 암호화된 값이므로 복호화 하여 암호화 되지 않은 값을 얻을 수 있다. 그리고 나서 해커의 프록시 서버에서 저장한 웹서버의 공개키로 그 값을 암호화 하여 웹서버 에게 전송한다. 클라이언트는 웹서버에 로그인 하게 되고 메일 검색 등 자신의 일을 수행 후 종료하게 된다.

4. SSL Man-in-the-Middle Proxy 해킹 방지 방법

SSL Man-in-the-Middle Proxy 해킹에 대응하기 위해서는 다음과 같은 방법들이 필요하다. 우선 사용자 또는 네트워크 관리자가 설정한 프록시 서버 설정 변경이 되었는지 확인하고 사용자의 의도와 다르게 변경된 프록시 서버로 데이터가 전송될 시에는 세션을 차단하는 방법이 있다. 프록시 설정이 되었는지의 여부를 판단하기 위해서는 패킷의 패턴을 분석하여 프록시로 데이터를 전송하는 지의 여부를 판단할 수 있다. 예컨대, 그림 3의 프록시 서버 연결에서 클라이언트에서 전송되는 패킷의 내용을 분석해 보면 그림 4에서 일반적인 접속과는 달리 프록시 서버로 데이터를 전송하는 것을 확인할 수 있고, 사용자나 관리자에 의해서 프록시 설정이 되어 있지 않았음에도 프록시 패턴이 감지되면 경고창과 함께 클라이언트에서 서버로 기밀정보가 전송되지 못하도록 세션을 차단해야 한다.

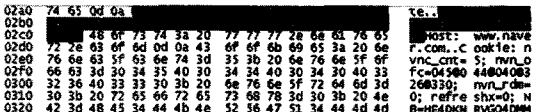

그림 3 프록시 서버 연결

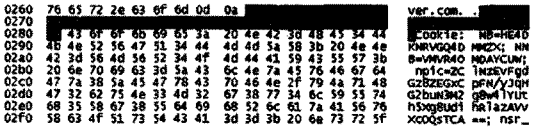

그림 4 일반적인 연결

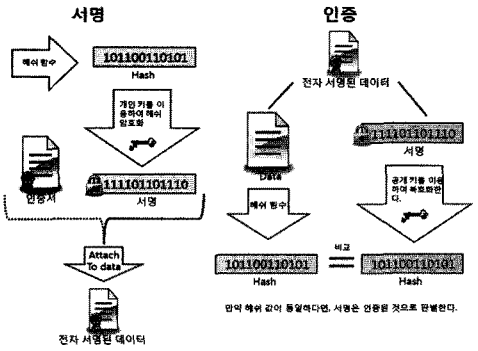

그림 5 전자서명과 검증

만약, 사용자나 네트워크 관리자에 의해 설정된 프록시 서버라면 RSA 알고리즘을 이용한 전자서명 방법을 이용하여 프록시 서버를 인증한 뒤에 기밀정보를 전송하여 프록시 서버를 이용한 MITM을 방지할 수도 있다.

4.1 인가되지 않은 프록시 설정 방지

프록시 변조 해킹을 방지하기 위한 방법으로는 클라이언트의 레지스트리 프록시 설정 값이 악의적인 프로그램에 의해 변조되었는지를 판단하기 위하여 해시 함수를 이용한 무결성 검증 기능이 필요하며 또한 보안 솔루션에서는 중요한 데이터를 인터넷 통해 전송 시 악성 프로그램이나 스크립트에 의해 사용자의 의도와 달리 클라이언트가 프록시 서버 설정이 변경 되었는지를 확인하여 중요 정보가 전송되기 전에 경고 메시지를 띄워 주는 기능을 가져야 한다.

본 논문에서 제시하는 과정을 구체적으로 설명하자면 다음과 같다. 클라이언트가 서버에게 서비스를 요청하면 서버는 클라이언트에게 서비스를 제공하고 클라이언트가 현재 프록시 설정이 되어 있다면 프록시 설정이 되어 있다는 경고 메시지를 띄워 주어 사용자에게 해킹의 위험성이 있음을 인지시킨 후에 임의의 프록시 서버를 사용할 수 있게 한다. 하지만 현재 대다수 보안 시스템은 제3장에서 예시한 바와 같이 프록시 경유의 위험성에 대해서는 간과하고 있다. 본 논문에서 제시하고 있는 프록시 설정 무결성 검증과 중요 정보 전송 시에 프록시 사용 설정 경고를 나타나게 하고 임의로 프록시 설정을 하지 않는 이상 기본적으로 프록시를 경유하지 않도록 한다면 MITM 공격으로 인한 피해를 예방할 수 있다.

4.2 인증 프록시 서버

RSA 공개키 알고리즘을 이용하여 프록시 서버를 인증하는 방법이다. 이를 이용하면 공격자에 의해 클라이언트의 프록시 서버 설정이 변경될 시에는 검증 알고리즘을 이용하여 관리자에 의해 인증된 프록시 여부를 확인 후에 인증되지 않은 프록시로 판단되면 세션을 차단한다. 관리자에 의해 프록시를 사용하는 환경에서는 그림 5와 같이 프록시는 데이터를 해시 함수로 다이제스트 시킨 후에 그 값을 개인키로 암호화시킨다. 개인키로 암호화된 값은 프록시의 공인인증서와 함께 클라이언트로 전송이 되고 클라이언트는 전송받은 데이터를 프록시의 공인인증서에서 추출한 공개키로 서명값을 검증한다. 검증된 프록시의 진위 여부가 확인되면 클라이언트는 중요 데이터를 프록시로 전송한다. 본 논문에서 제안하는 프로토콜의 기호는 표 1과 같다.

프록시 인증 프로토콜은 그림 6과 같고, 프로토콜이 모두 정상적으로 수행되면 클라이언트 C와 변조되지 않은 프록시 서버 S는 동일한 키 $k=(e^b)^a \bmod n = (e^a)^b \bmod n = e^{ab} \bmod n$을 계산하게 되고, 동일한 세션 키

표 1 기호

기호	설명
θ	보안 변수
PKE = (PK,PE,PD)	공개키 암호 알고리즘. PK는 키 생성 함수로 θ를 입력하여 개인키와 공개키 쌍인 (e,d)를 생성, $PE_e(m)$는 암호화 알고리즘으로 e를 사용하여 평문 m에 대한 암호문 c를 출력. $PD_d(c)$는 복호화 알고리즘으로 d를 사용하여 암호문 c에 대한 평문 m을 출력.
Σ = (Gen, Sign, Verify)	서명 알고리즘. Gen은 키 생성 함수로 θ를 입력하여 전자서명용 키와 검증용 키 쌍인 (d,e)를 생성. $Sign_d(m)$은 서명 생성 알고리즘으로 d를 사용하여 평문 m에 대한 서명 σ를 생성. $Verify_e(m,\sigma)$는 서명 검증 알고리즘으로 e를 사용하여 만약 평문 m에 대한 서명 σ가 올바르다면 1을 출력하고, 그렇지 않다면 0을 출력.
c, s	Client C와 Proxy P의 ID
$b \xleftarrow{R} [1,p]$	1에서 p사이의 수로부터 임의로 선택된 b
$Cert_s$	S의 공개키 인증서
H	해시 함수

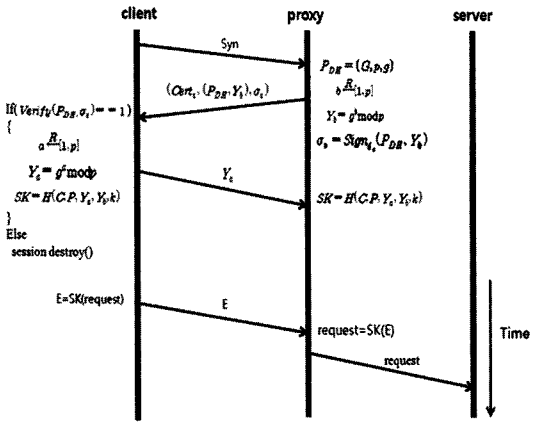

그림 6 프록시 인증 프로토콜

Step 1. 프록시 서버 S는 Diffie-Hellman[10] 파라미터인 $P_{DH}=(G,p,g)$를 생성한다. G는 소수 p를 모수로 갖는 순환군이고, g는 G의 생성자이다. 프록시 서버 S는 $[1,p]$ 내에서 랜덤하게 b를 선택하여 $Y_b=g^b \bmod p$를 계산한다. RSA 개인키 d_s를 이용해 서명 값 $\sigma_s=Sign_{d_s}(P_{DH}, Y_b)$를 계산 후, 클라이언트 C에게 $(Cert_s,(P_{DH},Y_b),\sigma_s)$를 전송한다.

Step 2. 클라이언트 C는 $Cert_s$의 유효성을 검증 후, 유효하다면 $Cert_s$에서 공개키를 추출하여 서명 σ_s를 검증하여 $Verify(P_{DH},\sigma_s)=1$이 출력되면 변조되지 않은 프록시 서버 S 임을 확인하고 서명이 유효하지 않다면 세션을 종료한다.

Step 3. 클라이언트 C는 $[1,p]$내에서 랜덤하게 a를 선택하여 $Y_a=g^a \bmod p$를 계산하고, 프록시 서버 S에게 Y_a를 전송한다. Y_a

Step 4. 클라이언트 C와 프록시 서버 S는 대칭키 암호 시스템에서 데이터 암호화 시 사용 가능한 형태의 세션 키 $SK=H(C,P,Y_a,Y_b,k)$를 계산한다.

Step 5. 클라이언트 C에서 프록시 서버 S 로 전송되는 값들은 세션 키 SK로 암호화 되어 전송된다.

SK=$H(e^{ab})$를 공유한다. 공유된 키를 세션에서 가지고 데이터 전송시 암호화된 패킷을 주고받기 때문에 클라이언트와 프록시 서버 사이에 MITM 공격을 방지할 수 있다.

5. 결론 및 향후 연구

악성 프로그램이나 스크립트 등에 의해서 클라이언트의 설정 변경이 되면 의도하지 않는 프록시 서버로 데이터를 전송하여 MITM 해킹의 위험이 매우 큰 것에 비해 인지도는 매우 낮다. 그러므로 클라이언트에서 프록시 서버로의 연결 절차는 보안성이 충분히 고려된 후 제작되어야 함에도 불구하고 클라이언트는 프록시 서버에 대한 인증 절차 없이 프록시 서버로의 데이터 전송이 이루어지고 있기 때문에 인터넷 뱅킹 시스템을 사용하는 클라이언트는 자신의 기밀정보들이 이전에 설정해 놓은 프록시 서버로 전송이 되고 있는지 아닌지 판단을 할 수 없었다. 본 논문에서는 이러한 프록시 설정의 취약성에 대해서 분석하고 프록시 변조 해킹을 방지하기 위하여 상용 뱅킹 시스템 보안 솔루션이 갖추어야 할 클라이언트 프록시 설정에 대한 보안 기능을 제시 하였고, 근본적인 프록시 해킹 차단을 위해 공개키 알고리즘을 이용한 프록시 인증 프로토콜을 제시하였다.

이에 따라 본 논문에서 제안한 웹 프록시 서버 대한 인증절차 뿐만 아니라, 다양한 분야에서 쓰이고 있는 프록시에 적용할 수 있는 인증 프록시들에 대한 응용 연구가 차후 필요할 것으로 본다.

참 고 문 헌

[1] Dierks, T., and Rescorla, E., "The TLS Protocol Version 1.1," *RFC 4346*, 2006.

[2] Oppliger, R., Hauser, R., and Basin, D., "SSL/TLS Session-Sware User Authentication Revisited," In *Proc. COMPSEC*, pp.64-70, 2008.

[3] Oppliger, R., Hauser, R., and Basin, D., "SSL/TLS Session-Aware User Authentication," *Computer & Communication*, vol.41(3), pp.59-65, 2008.

[4] Burkholder, P., "SSL Man-in-the-Middle Attacks," *SANS Institute*, p.15, 2002.

[5] Bringer, J., and Chabanne H., "Trusted-HB: A Low-Cost Version of HB Secure Against Man-in-the-Middle Attacks," *IEEE Transactions on Information Theory*, vol.54(9), pp.4339-4342, 2008.

[6] Klein, A., "HTTP Response Splitting, Web Cache Poisoning Attacks, and Related Topics," *Sanctum Inc.*, p.31, 2004.

[7] Liu, A., Yuan, Y., Wijesekera, D., and Stavrou, A., "SQLProb: a Proxy-Based Architecture Towards Preventing SQL Injection Attacks," In *Proc. SAC* pp.2054-2061, 2009.

[8] Oppliger, R., and Gajek, S., "Effective Protection Against Phishing and Web Spoofing," In *Proc. CMS*, vol.3677, pp.32-41, 2005.

[9] Adelsbach, A., and Gajek, S., Schwenk, J., "Visual Spoofing of SSL Protected Web Sites and Effective Countermeasures," In *Proc. ISPEC*, pp.204-216, 2005.

[10] W. Diffie and M.E. Hellman, "New directions in cryptography," *IEEE Transactions on Information Theory*, vol.22, no.6, pp.644-654, 1976.