Single Sign-on에 적용 가능한 인증 모델에 관한 연구

서대희, 이임영
*순천향대학교 정보기술공학부

Dae-Hee Seo, Im-Yeong Lee
* Division of Information Technology Engineering. SoonChunHyang University

요약

인터넷의 급속한 확산으로 인해 사용자들은 다양한 서비스 제공을 받고 있다. 그러나 사용자들 다양한 어플리케이션 서비스를 받기 위해서 각각 다른 ID와 비밀번호를 기억해야 하는 불편을 감수해야만한다. 이러한 비효율적이면서 보안적으로 많은 취약성을 가지고 있어 제안된 시스템이 SSO이다. SSO는 사용자와 관리자에게 효율성과 보안성을 동시에 높여 줄 수 있는 시스템이지만, 다양해져가는 인터넷 환경 속에서 여전히 문제점을 가지고 있다. 따라서 본 논문에서는 SSO에 적용 가능한 브로커 인증 모델을 제시함으로써 SSO를 실제 구성시 기존 모델이 가지고 있는 보안적인 취약성과 효율성을 높이고자 하였다.

1. 서 론

최근 인터넷의 급속한 확산과 더불어 대중화된 웹 서비스에서 사용자들은 수많은 웹 사이트에 가입하여 여러 가지 편리하고 효율적인 서비스를 제공받고 있다.

이러한 현상은 기업이 점점 성장하면서 시스템 및 어플리케이션의 수와 종류도 다양하게 증가해가기 때문이다.

그러나 이러한 복잡성으로 인해서 사용자들은 시스템에 액세스 할 때마다 각각 다른 ID와 비밀번호를 기억해야 하는 불편과 함께 관리자로서는 모든 ID와 비밀번호를 관리해야 된다는 문제점이 발생하게 된다.

따라서 이를 해결할 수 있는 시스템이 SSO(Single Sign-On)이다. SSO는 사용자 뿐만 아니라 관리자에게 비용 및 편리성을 제공할 수 있는 매우 강한 솔루션이라 할 수 있다. SSO 서비스에서 가장 중요한 요소는 사용자의 Identity를 인증모델에 있으며 이는 전체적인 SSO의 효율성과 안전성에 매우 큰 영향을 미치게 된다[1~3].

이에 본 논문에서는 SSO의 인증 모델 중에서 브로커 모델을 기반으로 효율성과 보안성을 향상시킨 브로커 인증 모델을 제시하였다. 본 논문의 2장에서는 SSO 인증모델 중 브로커 모델에 대해 논한 뒤 실제 SSO 인증 모델이 가져야 하는 보안적 요구사항을 3장에서 제시하고자 한다. 4장에서는 기존 브로커 모델이 가지는 단점과 3장에서 제시한 보안 요구사항을 만족하는 안전하고 효율적인 브로커 모델을 제시한다. 5장에서는 4장에서의 제안방식을 분석한 뒤 6장에서 결론을 맺고자 한다.

2. 브로커 인증 모델

SSO는 사용자가 한번 Sign-On 한 뒤 모든 서비스를 이용할 수 있도록 하는 기술로서 사용자의 패스워드 분실 처리에 관련된 자원 및 비용을 감소 시키면서 중앙 집중식 인증/권한 관리를 통해 관리 생산성을 증대시켜 사용자 뿐만 아니라 관리자의 요구사항도 만족할 수 있는 방법이다.

SSO를 구성하기 위해서는 사용자의 신원을 확인하는 인증 과정과 인증 과정을 거친 후 해당 자원에 대한 접근이 가능하도록 하는 인정 과정 그리고 인증과 인정 과정의 통합 관리 뿐만 아니라 사용자와 계정 및 패스워드의 통합 관리를 들 수 있다. 이상의 3가지 과정 중에서도 사용자의 신원을 확인하는 것은 매우 중요하다 할 수 있으며 이와 관련된 대표적인 인증 모델이 브로커 인증 모델이다[4~5].

브로커 인증 모델의 특징은 다음과 같이 몇가지로 요약해 볼 수 있다.

- 구현 : 프로그램 수정
- 관리 : 중앙 집중형 관리
- 보안성 : 모든 통신의 암호화가 필수
- 사용 편리성 : 클라이언트 사용법과 사용자가 자신을 인증하는 방법을 숙지하고 있어야 함.

3. 브로커 인증 모델의 보안 요구사항

신뢰받지 않은 네트워크를 통해 사용자는 자신의 중요한 인증 정보를 전송하기 위해서는 다음과 같은 몇가지 보안 요구사항이 요구된다[1~6].

• 기밀성 : 전송되는 데이터는 암호화를 통해 제 3자로의 공격으로부터 안전성을 유지할 수 있어야 한다.

• 무결성 : 신뢰받지 않은 네트워크를 통해 사용자의 인증 정보가 전송될 경우 공격자가 이를 위조하거나 변조할 수 있으므로 이에 대한 전송 데이터의 무결성이 반드시 요구된다.

• 부인봉쇄 : 인증 모델에 참여하는 모든 객체들에 대해 데이터의 전송 및 수신에 대한 부인을 막을 수 있는 보안 서비스가 제공되어야 한다.

• 사용자의 사전등록 : 사용자의 사전 등록 과정에서 사용자가 생성한 비밀스러운 값이 공개되지 않으면서 이를 검증할 수 있어야 한다.

• 익명 사용자 : 익명 사용자의 경우 특별한 등록 과정 없이도 자신의 공개된 정보로 서비스 이용이 가능해야 하며, 이를 검증할 수 있어야 한다.

4. SSO에 적용 가능한 브로커 인증 모델

본 논문에서는 클라이언트와 인증 서버 그리고 Kerberized Server를 중심으로 이루어지는 브로커 인증 모델을 제시하고자 한다. 제안방식은 다음과 같은 3개의 구성 객체로 이루어진다.

• 인증 서버 : 브로커 인증 모델에 참여하는 모든 객체들의 공개 고유값을 공개 디렉토리에 공개하고, 고유 ID와 Client/Server ticket을 생성하여 클라이언트와 Kerberized server에게 이를 전송하는 객체

• 클라이언트 : 인증 서버로부터 Client ticket과 고유 ID를 발급받고 이를 기반으로 Kerberized server의 어플리케이션 서비스를 제공받고자 하는 객체

• Kerberized server : 여러 어플리케이션 서비스를 관할하는 통합 객체로서 인증 서버로부터 Server ticket과 고유 ID를 발급 받고 이를 기반으로 클라이언트에 어플리케이션 서비스를 제공하는 객체

4.1 시스템 계수

다음은 SSO에 적용 가능한 브로커 인증 모델을 위한 시스템 계수를 기술한다.

* (클라이언트 : c, 인증 서버 : a, Kerberized server : k)

p_* : *가 생성한 소수

($512bit \le Length \le 1024bit$, $2^{L-1} < p_* < 2^L$ 을 만족하는 소수)

q_* : $p-1$을 만족하는 수

r_* : *의 객체가 생성한 의사난수

n : RSA의 모듈러

g : Z_n상에서 최대 위수를 갖는 원소

ID_* : *의 information data

E : 대칭키 암호 알고리즘

T_* : *가 생성한 타임 스탬프

H : 안전한 해쉬 알고리즘

M_* : 세션키 C_{ck}로 암호화된 서비스 접속 메시지

$V_{*@}$: *에서 @으로 전송되는 암호화된 값

m_a : 인증서버의 인증 메시지

Server Ticket : 인증 서버가 생성한 서비스 ticket으로써 다음과 같은 내용을 포함하고 있다. (Ticket 요청자, ticket 응답자, 생성시간, 유효기간)

Client ticket : 인증서버에서 생성한 클라이언트 ticket으로써 다음과 같은 내용을 포함하고 있다. (Server Ticket + (ticket 요청자, ticket 응답자명, 생성시간, 유효기간))

4.2 제안 방식 단계

본 제안 방식은 인증서버의 공개키 P_a가 모든 객체에게 공개되어 있다는 가정을 기반으로 다음과 같은 단계로 진행된다.

[Step 1] 초기 등록 단계 : 클라이언트와 Kerberized Server는 인증서버에 자신의 고유 공개값을 등록하는 단계

[Step 2] 서버와 클라이언트 정보 전송 단계 : Step 2는 서버와 클라이언트에 전송할 ticket 및 고유 ID를 인증서버가 생성하여 이를 전송하는 단계

[Step 3] 클라이언의 서비스 요청 단계 : 서비스 요청 단계는 클라이언트의 고유 ID와 Client ticket를 확인하고 세션키를 이용해 암호화된 메시지를 전송하는 단계

[Step 4] Kerberized Server의 ticket 및 세션키 획득 단계 : Kerberized Server는 인증 서버에서 전송된 정보를 통해 자신의 고유 ID와 세션키를 획득하여 Server ticket을 획득하는 단계

[Step 5] 클라이언트의 서비스 인증 단계 : Kerberized Server는 클라이언트로부터 전송

된 정보를 기반으로 세션키를 통해 암호화된 메시지를 복호화 하여 해당 어플리케이션 서비스에 연결을 승인하는 단계

4.3 프로토콜

[Step 1] 초기 등록 단계

클라이언트와 Kerberized Server는 인증 서버에 고유 공개값을 등록하기 위하여 다음과 같은 계산을 수행하여 (P_k, T_k), (P_c, T_c)를 인증서버에 전송한다.

- 클라이언트 : 클라이언트는 r_c ($r_c \in (1,\ldots,p_c-2)$)를 선택한 후 공개된 시스템 계수를 이용하여 P_c를 생성한다.

$$P_c = g^{r_c} \bmod n$$

- Kerberized server : 인증서버는 r_k ($r_k \in (1,\ldots,p_k-2)$)를 선택 후 공개된 시스템 계수를 이용하여 P_k를 생성한다.

$$P_k = g^{r_k} \bmod n$$

[Step 2] 서버와 클라이언트 정보 전송 단계

인증 서버는 전송된 (P_k, T_k), (P_c, T_c)를 공개 디렉토리에 저장하고 다음과 같은 과정을 수행한다.

- 클라이언트에서 전송된 (P_c, T_c)를 확인한 뒤 클라이언트의 고유 ID_c와 *Client ticket*를 생성한 뒤 다음을 계산하여 클라이언트에게 a_{ac}, T_a, V_{ac}, ID_c, h_{ac}를 전송한다.

$$K_{ac} = H(P_c^{r_a} \| ID_c) \bmod p$$

$$a_{ac} = g^{r_a} \bmod p$$

$$S_{ac} = \frac{r_a}{(Z_{ac}+r_a)} \bmod q$$

$$Z_{ac} = KH_{a_{ac}}(Clientticket \| P_a) \oplus H(m \| P_c) \bmod p$$

$$h_{ac} = H(a_{ac} \| T_a \| C_{ac})$$

$$V_{ac} = E_{K_{ac}}(m_a \| Clientticket \| a_{ac})$$

- 인증 서버는 Kerberized server의 고유 ID_k와 *Server ticket*을 생성한 뒤 다음을 계산하여 Kerberized server에게 a_{ak}, T_a, V_{ak}, ID_k, h_{ak}를 전송한다.

$$K_{ak} = H(P_k^{r_a} \| ID_k) \bmod p$$

$$a_{ak} = g^{r_a} \bmod p$$

$$S_{ak} = \frac{r_a}{(Z_{ak}+r_a)} \bmod q$$

$$Z_{ak} = KH_a(Serverticket \| P_a) \oplus H(m \| P_k) \bmod p$$

$$h_{ak} = H(a_{ak} \| T_a \| V_{ak})$$

$$V_{ak} = E_{K_{ak}}(m_a \| Serverticket \| a_{ak} \| ID_c)$$

[Step 3] 클라이언의 서비스 요청 단계

클라이언트는 Step 2에서 전송 받은 a_{ac}, T_{ac}, V_{ac}, ID_c, h_{ac}에서 다음과 같은 검증 과정을 거쳐 전송 데이터에 대한 검증을 수행한다.

$K_{ac}' = H(a_c^{r_c} \| ID_c) \bmod p$, $K_{ac}' \stackrel{?}{=} K_{ac}$이면 전송된 V_{ac}를 복호화하여 m, T_a, *Clientticket*, a_{ac}를 확인 한 뒤 이를 기반으로 a_{ac}를 검증한다.

$a_{ac}' = a_{ac}$이면 $Z_{ac}' = Z_{ac}$

이상의 검증 과정을 거쳐 클라이언트는 인증 서버로부터 전송되어온 *Client ticket*을 획득한다. Client ticket을 획득한 클라이언트는 서버에 전송하고자 하는 메시지를 세션키 C_{ck}를 다음과 같이 생성한다.

$$C_{ck} = H(m_a \oplus ID_c) \bmod p$$

이후 클라이언트는 세션키 C_{ck}로 메시지를 암호화 한 뒤 V_{ak}, T_{ck}를 Kerberized server에 전송하여 서비스를 요청한다.

[Step 4] Kerberized Server의 ticket 및 세션키 획득 단계

Kerberized server는 인증 서버로부터 전송받은 a_{ak}, T_{ak}, V_{ak}, ID_k, h_{ak}를 이용하여 다음을 검증한다.

$$K_{ak}' \stackrel{?}{=} K_{ak},\ a_{ak}' \stackrel{?}{=} a_{ak},\ Z_{ak}' \stackrel{?}{=} Z_{ak}$$

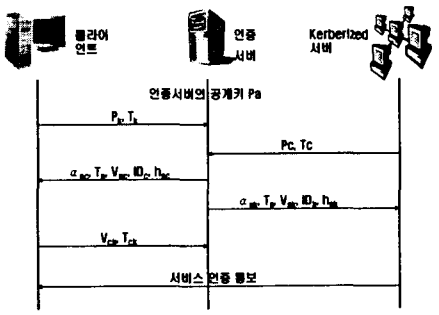
이상의 내용이 올바른 경우 클라이언트와의 세션키 C_{ck}를 생성하고 Server ticket을 획득한다.

$$C_{ck} = H(m_a \oplus ID_k) \bmod p$$

[Step 5] 클라이언트의 서비스 인증 단계

Step 3에서 클라이언트로부터 전송된 M_c, T_{ck}에서 V_{ak}를 Step 4에서 생성한 세션키 C_{ck}로 복호화 하여, 클라이언트가 접속하고자 하는 어플리케이션 서비스에 연결한다.

이상의 프로토콜 진행을 통해 SSO에 적용 가능한 브로커 인증 모델을 제안하였다. 이상의 프로토콜 진행을 다음과 같이 그림 1로 표현할 수 있다.



(그림 1) SSO에 적용 가능한 브로커 인증 모델

5. 제안방식 분석

본 장에서는 4장에서 제시했던 제안 방식을 3장에서 기술한 보안 요구사항을 기반으로 분석하고자 한다.

제안 방식은 보안 요구사항에 기반하여 다음과 같은 특징을 가지고 있다.

- 기밀성 : 브로커 인증 모델의 참여 객체들은 이산대수 문제에 근거한 공개 정보와 서버의 인증 메시지와 *ID*를 이용한 세션키 설정으로 제 3자의 공격으로부터 안전한 데이터 전송이 가능하다.
- 무결성 : 신뢰받지 않은 네트워크를 통해 사용자의 인증 정보가 전송될 경우 안전한 해쉬 함수와 타임스탬프를 통해 무결성 서비스를 제공할 수 있다.
- 부인봉쇄 : 인증 모델에 참여하는 모든 객체들은 자신만의 비밀 의사난수인 r을 선택하여 서비스 정보가 전송된다. 선택된 의사난수는 공개키를 생성할 때 사용된다. 따라서 각 클라이언트 마다 고유의 공개키가 존재하고 이를 기반으로 전송 데이터의 부인봉쇄가 가능하다.
- 사용자의 사전등록 : 사용자의 사전 등록 과정에서 사용자가 생성한 비밀스러운 값 r이 공개되지 않으면서도 이를 기반으로 생성한 P의 값을 이용해 자신이 비밀스럽게 소유한 r를 검증할 수 있다.
- 익명 사용자 : 익명 사용자가 서비스를 제공받고자 한다면 인증 서버에서 할당해주는 고유 ID와 익명 사용자가 전송한 공개정보에 의해 익명 서비스가 가능하다.

6. 결론

최근 인터넷의 급성장과 더불어 다양한 서비스가 사용자에게 제공되고 있다. 그에 따라 사용자는 자신이 관리해야 할 고유 인증 정보(ID, 패스워드)를 지속적으로 기억해야 함은 물론이고 새로운 서비스를 받고자 할 경우에도 많은 불편함을 가지고 있다. 인터넷 서비스 관리자의 입장에서는 분산되어 있는 사용자들을 통합적인 관리가 매우 어려운 실정이다.

따라서 본 논문에서는 SSO에 적용 가능한 인증 모델을 제시하였다. 이는 SSO에서 가장 중요한 부분중의 하나인 인증 부분에서 필요한 여러 가지 보안 요구사항을 제시하고 이를 만족할 수 있는 사용자의 편리성과 관리자의 중앙집중식 관리를 통해 기존의 SSO 인증 모델중의 하나인 브로커 모델의 보안적인 취약점 뿐만 아니라 사용자의 초기값 전송에서 익명 사용자의 보안 문제까지 해결하였다.

향후 제안 방식을 멀티 에이전트를 결합한 혼합 형태의 인증 모델을 제시하고자 한다.

7. 참고 문헌

[1] http://technet.oracle.co.kr/docs/oracle78/network23x/NWANO233/ch1.htm

[2]. B.C. Neuman, Theodore Ts'o. Kerberos, "An Authentication Service for computer Network," IEEE Communication, 32(9) : 33-38, September, 1994

[3]. 최용락. 소우영, 이재광, 이임영 "통신망정보보호", 그린출판사, pp343-393, 2001

[4]. RFC 1510, Public key Cryptography for Initial Authentication in Kerberos, draft-ietf-cat-kerberos-pkinit-14.txt.

[5]. http://www.krnet.or.kr/

[6]. 이만영, 김지홍, 류재철, 송유진, 염홍렬, 이임영 "전자상거래 보안 기술", 생능출판사 1999.8.

[7]. Alfred J. Menezes, Paul C.van Oorschot, Scott A. Vanstone "HANDBOOK of APPLIED CRYPTOGRAPHY", CRC, 1996.11.