해성을 추가한 안전한 IV 전송 알고리즘

이영지, 김태윤, 현은실, 박남섭
고려대학교 컴퓨터학과
e-mail : {yjlee, tykim, eunsil, nspark}@netlab.korea.ac.kr

IV safe transfer algorithm adding hashing

Lee, young-ji, Kim, tai-yun
Dept. of Computer Science, Korea University

요 약

IPSec(IP Security)은 데이터가 공개적으로 전송되는 네트워크에서 데이터에 암호화와 인증, 무결성을 제공하기 위해 사용되는 프로토콜이다. IPSec 안에는 여러 프로토콜이 있는데, 그 중에 실제 패킷에 암호화와 인증, 무결성을 추가해 전달하기 위해서는 ESP(Escapsulation Security Payload)라는 프로토콜이 사용된다. 이 ESP는 패킷을 암호화하기 위해 DES-CBC 모드를 사용하는데, 여기에서 IV(Initialization Vector) 값이 쓰인다.

이 값은 패킷 복호화를 하기 위해 공개적으로 전달이 되기 때문에 공격자에 의해 공격 당할 위험이 많다.

본 논문에서는 IV 공격을 방지하기 위해 IV의 값을 해쉬 함수를 통해 한번 해성을 한 다음에, IV 값을 안전하게 전달하는 방법을 제시하고자 한다.

1. 서론

네트워크 상으로 전송되는 중요한 데이터들에 공개적으로 전송될 때 그 데이터들에 대한 안정성과 인증이 필요하게 되었다.

IPSec(IP Security)은 인터넷과 같은 공개적인 공중 통신망에서 데이터를 전송할 때 데이터를 보호하기 위한 프로토콜이다. IPSec 안에는 기본적으로 여러 프로토콜이 포함된다.

AH(Authentication Header)는 실제 데이터 패킷에 무결성과 인증을 제공해 주는 프로토콜이고, AH의 기능에 암호화를 더한 기능의 ESP(Escapsulation Security Payload)가 있다. 그리고 두 통신 호스트 사이에 데이터 패킷을 주고받기 위한 여러 가지 사항들에 대한 내용을 결정하는 SA(Security Association)를 합성하기 위한 IKE(Internet Key Exchange) 프로토콜이 있다.

본 논문에서는 이 중에서도 ESP에서 받기 쉬운 공격인 'IV 공격'을 방어하기 위한 해결책을 제시한다.

2. 관련 연구

- ESP

ESP는 IP 패킷에 대해 데이터 무결성, 인증, 암호화, replay 공격에 대해 방어 등의 기능을 제공하는 프로토콜이다. 원래의 IP 패킷과 헤더 사이에 새로운 ESP 헤더를 삽입하게 된다. ESP는 다음과 같이 헤더와 트레일러로 구성이 된다. 트레일러는 암호화가 되지만, 헤더 부분은 암호화가 되지 않은, 공개된 상태로 전송이 된다.

<table>
<thead>
<tr>
<th>IP header</th>
<th>ESP header</th>
<th>protected data</th>
<th>ESP trailer</th>
</tr>
</thead>
</table>

encrypted

authenticated
ESP에서 쓰이는 모든 암호 알고리즘은 CBC (Cipher Block Chaining) 모드를 사용한다. CBC 모드로 암호화 하려면 데이터가 암호 복록 사이즈의 배수의 길이가 되어야 한다. 이것은 데이터의 끝에 패딩(padding)을 붙여서 해결한다. 이 패딩은 데이터와 같이 암호화가 되어 송신자로부터 복호화 된다.

- SA (Security Association)는 IPSec를 통해 데이터를 전송하는 수행 과정에 필요한 키, 패킷 안전성, 전송 형태 등 여러 가지 사항들을 결정한다. SA는 두 동신 개체 사이의 협상이므로 실제 패킷을 전달하기 전에 모든 협상이 끝나야 한다.

SA는 단방향이므로, 나가는 패킷과 들어오는 패킷에 대해 각각의 SA를 사용한다. 이런 작업은 IKE (Internet Key Exchange)를 통해서 이루어진다. IKE는 공유된 비밀 파라미터와 인증된 키들을 생성한다.

SA에는 두 단계가 있다. 우선 암호화된 데이터를 전송할 때 필요한 사항을 가진 SA (quick mode)와 이 SA를 또 설정하기 위한 SA에 대한 SA (main mode)가 있다.

- IV

CBC 모드에서의 암호는 IV (Initialization Vector)를 필요로 하는데, 이것은 데이터 페이드 밀드에 랜덤하게 바이트로 포함된다. 이것은 암호화 알고리즘에 따라 달라지는데, 예를 들어 IPSec에서 스 이하는 DES-CBC 모드에서는 보호되는 데이터 밀드의 첫 번째 8 비트가 IV가 된다.

IV는 암호화된 데이터를 송신자가 해쉬 함수를 통해 다시 암호화하는 데 사용된다. 이는 ESP에서 IV가 포함되는 패킷을 나타낸 그림이다.

<table>
<thead>
<tr>
<th>Security Parameters Index (SPI)</th>
<th>Sequence Number Field</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initialization Vector</td>
<td>Protected Payload Data (Variable)</td>
</tr>
<tr>
<td>Optional Padding (0~255)</td>
<td>Pad Len. Next Header</td>
</tr>
</tbody>
</table>

<그림 2 ESP에 의해 보호되는 IP 패킷>

이 중에서 Payload Data가 아닌 암호화가 되고 IV 값은 그대로 전송이 된다. 공격자는 네트워크에서 패킷을 가로截取 다음에 이 IV 값을 변경시키게 된다.

• 해쉬 함수 (hash function)

해쉬 함수는 임의의 입력 비트열에 대하여 일정한 길이의 안전한 출력 비트열을 내는 함수로써 주어진 해쉬 코드에 대하여 이 해쉬 코드를 생성하는 데이터 스트림을 찾아내는 것은 계산량 불가능하고, 주어진 데이터 스트림에 대해 같은 해쉬 코드를 생성하는 또 다른 데이터 스트림을 찾아내는 것은 불가능하다는 성질을 만족하는 함수이다.

해쉬 함수 h는 만족시켜야 하는 내 가지 사항이 있는데, 우선 함수 h는 입력의 크기가 원하는 입력 M에 적용할 수 있어야 하고, 입력값 M에 대하여 일정한 길이의 출력값 H = h(M)을 볼 수 있어야 한다. 함수 h와 출력값 H가 주어졌을 때 H = h(M)을 계산하기 쉬워야 하고, 입력값 M을 구하는 역계산이 불가능해야 한다.

3. IV 공격

IV (Initialization Vector)는 3DES-CBC 모드에서 ESP의 데이터에 대한 암호화를 제공하는 초기 값이다. ESP의 처음 패킷은 IV와의 XOR를 통해 암호화가 이루어진다. IV는 ESP 패킷에서 암호화되어 있고, 공개된 것으로 보이지 않기 때문에 공격당할 위험이 많다. 이렇게 IV를 변경시켜서 원하는 것을 IV 공격 (IV attack)이라고 한다. 이 공격은 CBC 암호에서 IV를 변경하여 인증되지 않은 IV를 사용한다. 공격자는 그 변경된 IV를 사용하여 데이터를 복호화할 수 없게 만들고, 또 복호화된 정보의 첫 번째 블록을 바꿀 수도 있다.

IV 공격은 특히 IPSec에서 문제가 된다. 

첫째, IPSec에서는 CBC 모드에서의 블록 암호를 사용하지 않고 있다. 블록 암호의 데이터를 일정한 크기의 블록 단위 끝으로 분리해서 암호화하는 것이다. 그렇기 때문에 IV가 손상이 되면 그 블록 안의 모든 데이터를 다시 빌리게 된다. 게다가, 이 알고리즘의 거의 대부분은 공개되어 있어 공격에 전공하는 IV를 사용하도록 하고 있다. IV는 ESP 데이터를 복호화 하는데 필요하기 때문에 IV값을 조작한 바뀌도 송신자는 원하는 데이터를 얻을 수가 없다.

이와 같은 문제점 때문에 IV 값을 안전하게 전송할 수 있는 방법이 연구되었다.
4. 제안된 IV 해제 알고리즘

본 논문에서 제안하는 알고리즘은 공개적으로 전송되는 IV의 값에 대해 해제하여 좀 더 안전한 상태로 전송을 하는 것이다.

SA에서는 호스트 간에 여러 가지 프로토콜과 패킷들에 대한 사항들을 결정하게 된다. IV를 해제한 값은 이 SA 중에 아직 사용되지 않은 필드들에 저장되어서 같이 전송이 되게 된다.

SA 과정에서 여러 가지 필드들이 전송이 되는데, 아래의 그림은 그런 필드들을 체이닝 방식(Chaining)을 통해 하나로 묶어서 전송하기 위한 헤더를 나타낸 것이다.

<table>
<thead>
<tr>
<th>Initiator Cookie</th>
<th>Responder Cookie</th>
</tr>
</thead>
<tbody>
<tr>
<td>Next Pay = 1</td>
<td>Major Ver.</td>
</tr>
<tr>
<td>Length</td>
<td>Message ID</td>
</tr>
<tr>
<td>Next Pay = 2</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>Situation</td>
</tr>
<tr>
<td>Next Pay = 0</td>
<td>0</td>
</tr>
<tr>
<td>Proposal # = 1</td>
<td>Proto ID = ESP</td>
</tr>
<tr>
<td>Spi</td>
<td>Trsfm # = 1</td>
</tr>
<tr>
<td>SA Attributes (See Attribute Payload Configuration)</td>
<td></td>
</tr>
<tr>
<td>Next Pay = 0</td>
<td>0</td>
</tr>
<tr>
<td>Trsfm # = 2</td>
<td>Transform ID</td>
</tr>
<tr>
<td>SA Attributes (See Attribute Payload Configuration)</td>
<td></td>
</tr>
</tbody>
</table>

#include <crypto.h>
/* 암호 헤더 파일을 추가한다.*/...

extern void IV_crypto(void);
/* 변수를 선언한다.*/

......
bool new_IV_set = TRUE;
/* hash 및 prf routines */

const struct hash_desc IV_hasher[IV_TIGER + 1] =
{
    { 0, NULL, NULL, NULL },
    /* no specified hasher */
    { MD5_DIGEST_SIZE,
      (void (*)(union hash_ctx *)) MD5Init,
      (void (*)(union hash_ctx *,
        const u_char *, unsigned int))
      MD5Update,
      (void (*)(u_char *, union
        hash_ctx *)) MD5Final},
    /* IV_MD5 */
    { SHA1_DIGEST_SIZE,
      (void (*)(union hash_ctx *))
      SHA1Init,
      (void (*)(union hash_ctx *,
        const u_char *, unsigned int))
      SHA1Update,
      (void (*)(u_char *, union hash_ctx *))
      SHA1Final},
    /* IV_SHA */
    { 0, NULL, NULL, NULL }
};

<표 1. IV 해제 알고리즘>

5. 결론 및 향후 연구 과제

본 논문에서 제안된 알고리즘은 IPSec의 ESP 패킷과 IV를 좀 더 안전하게 전송하고자 제안되었다. 제안된 알고리즘은 IV에 해제할 수 있도록 전송할 때 추가하는 값은 안전성과 함께 사용되지 않는 필드에 저장되어서 전송하고자 한다.

본 논문에서는 SA attributes의 필드에 추가해서 전송하고자 한다.
구를 통해서 많은 부분을 개선하는 것을 향후 과제로 삼고 있다.

참고문헌