IP Security Protocol Working Group (IPSEC)
INTERNET-DRAFT
draft-ietf-ipsec-ike-hash-revised-02.txt
Expires: 22 May 2001

T. Kivinen
SSH Communications Security

22 November 2000

Fixing IKE Phase 1 & 2 Authentication HASHs

Status of This memo

This document is a submission to the IETF IP Security Protocol (IPSEC) Working Group. Comments are solicited and should be addressed to the working group mailing list (ipsec@lists.tislabs.com) or to the editor.

This document is an Internet-Draft and is in full conformance with all provisions of Section 10 of RFC2026.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html.

Abstract

This document defines new method of calculating the authentication HASH of the IKE [RFC-2409] protocol. It fixes known problems with the IKE. The way the HASH is currently defined in the [RFC-2409] does not authenticate the generic ISAKMP [RFC-2408] header, nor does it authenticate any extra payloads inside phase 1 packets. This causes a security problem when using extra payloads as already defined in the IKE and DOI [RFC-2407] (vendor ID payload, INITIAL-CONTACT notification etc). There is also suggestion how to fix the Phase 2 authentication hashes so that they will also authenticate the generic ISAKMP header.

Table of Contents

1. Introduction

In the IKE [RFC-2409] protocol there is a clear security problem, because of the way the authentication HASH is calculated.

The HASH is defined in the [RFC-2409] like this:

```
HASH_I = prf(SKEYID, g^xi | g^xr | CKY-I | CKY-R | SAi_b | IDii_b )
HASH_R = prf(SKEYID, g^xr | g^xi | CKY-R | CKY-I | SAi_b | IDir_b )
```

The HASH does not include all payloads, nor it does not include generic ISAKMP [RFC-2408] header, which contains version numbers, exchange type etc.

2. Specification of Requirements

This document shall use the keywords "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED, "MAY", and "OPTIONAL" to describe requirements. They are to be interpreted as described in [RFC-2119] document.

3. Revised HASH Calculation

The new HASH is defined so that all the packet received and sent before are included in the HASH calculation. This also includes the packet currently being generated. The final authentication HASH is HASH of concatenation of HASHes of individiual packets. The reason for this is that quite a lot of implementations already calculate the HASH of the packet they are receiving just to detect retransmissions. This method also makes the memory consumption smaller.

Each packet HASH includes the ISAKMP generic headers, ISAKMP payload headers etc, i.e the exact bits sent to the wire from the beginning of the ISAKMP generic header to the end of the packet. The HASH includes the padding added because of the encryption. When the length of the packet (inside the ISAKMP header) is calculated in to the HASH, it MUST be set to the real length of packet including the padding. Packet is

added to the HASH as plaintext.

The authentication payloads (HASH or SIG) MUST be the last payload in

the packet, and when it is calculated to the authentication HASH it MUST have proper payload header, but its contents inside the payload MUST be all zeros, with proper length (either determined by the HASH algorithm or the public key used in the authentication).

So in the main mode the initiator HASH is calculated as follows:

HASH_I = prf(SKEYID, HASH(packet_1) | HASH(packet_2) | HASH(packet_3) | HASH(packet_4) | HASH(packet_5_template))

Where the HASH() is the negotiated hash algorithm. Note, that the initiator has to save the first packet he sends out, because he might not be able calculate the hash of the packet before he receives the responders packet and can find out the negotiated hash algorithm. Retransmission packets are not added to the HASH.

The packet_1 is the first packet initiator sends to the network (starting from the beginning of the generic header and continuing to the length specified in the ISAKMP header). Same goes for packets 2 to 4. The packet 5 template is special, because it is this packet we are currently sending out.

The HASH of the packet 5 template is calculated before encryption, but including the padding. The HASH/SIG payload MUST be in its place and MUST contain all zeros.

After the HASH of the packet has been calculated, then we calculate the actual HASH_I value. When the HASH_I has been calculated the place holder inside the packet is filled with the proper hash or signature, and the packet is encrypted before sent out.

When the responder is checking the HASH it first decrypts the packet_5_template and then it copies the HASH/SIG away and clears it from the packet. Then it calculates the exactly same HASH_I the initiator did, which can then be used authenticate the exchange (either direct comparison of the HASH, or signature verification).

In the main mode the responder HASH is calculated as follows:

HASH_R = prf(SKEYID, HASH(packet_1) | HASH(packet_2) | HASH(packet_3) | HASH(packet_4) | HASH(packet_5_template) | HASH(packet_6_template))

The packets 1 to 5 are identical to initiator case, i.e the SIG/HASH payload inside the payload 5 template contains zeros. The packet 6 template is similar than packet 5 template in the initiator case, i.e the HASH/SIG payload is in its place and must contain all zeros.

In the aggressive mode the HASH is calculated as follows:

HASH_I = prf(SKEYID, HASH(packet_1) | HASH(packet_2_template))

```
HASH_R = prf(SKEYID, HASH(packet_1) | HASH(packet_2_template) |
HASH(packet_3_template))
```

With same kind of processing of packet 2 and 3 than was for packets 5 and 6 in the main mode. Note, that the encryption of the final packet in the aggressive mode does affect the HASH, because there might be padding added to the packet 3 which must be then be included to the HASH.

4. Using of Revised HASH

The revised HASH is used for all new negotiations that are defined in the new IKE. This means that revised HASH is used if the phase 1 transform ID is specifying the next IKE version. Other new exchanges can define that they are also using the new revised HASH calculation method instead of the old HASH calculation method.

Each authentication method is exactly identical to the old ones, except the HASH_I and HASH_R are calculated as described in the section ``Revised HASH Calculation''.

In the signature modes the final SIG_I or SIG_R is the result of the negotiated digital signature algorithm applied to HASH_I or HASH_R respectively.

In the RSA Encryption mode the authentication of the other party takes place in the generation of the SKEYID, because to generate it correctly the other end must be able to decrypt the encrypted NONCE payload. Note that the ID and NONCE payloads are already encrypted using public key when they are calculated to the authentication HASH.

5. Fixing the Phase 2 authentication HASHs

For most of the Phase 2 exchanges the authentication hash is defined as follows:

HASH = prf(SKEYID_a, M-ID | rest of the packet after hash payload)

The new proposal for the authentication hash is:

HASH = prf(SKEYID_a, HASH(packet_template))

Where the packet_template is the whole packet before encryption, but after adding encryption padding. The HASH payload inside the packet MUST be in its place and its contents MUST be all zeros (generic payload header is properly filled in). Because the packet_template includes the generic packet header, which contains the message id field, there is no need to add that field to the hash separately.
This authentication hash SHOULD be used for all new exchange modes. I.e when new Phase 2 exchange mode is added it SHOULD use this kind of hash instead of old style hash, regardless of the phase 1 authentication style.

If the exchange contains multiple packets then the packets MUST be tied

together in the HASH calculation. This means that the HASH is calculated in the similar manner than in phase 1, i.e. the HASHes of the previous packets in the exchange are added before the HASH of the outgoing packet

template. For example the authentication HASHes for ficticious exchange having 4 packets are calculated as follows:

```
HASH(1) = prf(SKEYID_a, HASH(packet_1_template))
HASH(2) = prf(SKEYID_a, HASH(packet_1_template) |
HASH(packet_2_template))
HASH(3) = prf(SKEYID_a, HASH(packet_1_template) |
HASH(packet_2_template) | HASH(packet_3_template))
HASH(4) = prf(SKEYID_a, HASH(packet_1_template) |
HASH(packet_2_template) | HASH(packet_3_template) |
HASH(packet_4_template))
```

The packet templates are generated in same way than for one packet phase 2 exchange case.

For already existing phase 2 exchanges (quick mode, new group mode and informational exchange), this new hash MUST be used only and only if the ISAKMP SA was negotiated using the revised HASH authentication method. This will provide the backward compatibility with old implementations.

In the quick mode the HASH(2) and HASH(3) includes the nonce payloads, but if we include the complate hashes of 1st and 2nd packets to the HASH(2) and HASH(3) there is no need to add them separately to the HASH. Thus for the quick mode the new authentication HASH is defined to be:

```
HASH(1) = prf(SKEYID_a, HASH(packet_1_template))
HASH(2) = prf(SKEYID_a, HASH(packet_1_template) |
HASH(packet_2_template))
HASH(3) = prf(SKEYID_a, HASH(packet_1_template) |
HASH(packet_2_template) | HASH(packet_3_template))
```

6. Security Considerations

This document describes a way to fix the security problem inside the IKE. In the IKE defined in RFC2409 only some payloads are authenticated. This means that generic ISAKMP header (version numbers, exchange type, flags etc) and extra payloads (Notifications, Vendor ID, CERT, and CR payloads) are not authenticated. This document fixes that security problem.

7. References

[RFC-2408] Maughan D., Schertler M., Schneider M., Turner J., "Internet Security Association and Key Management Protocol (ISAKMP)", November 1998.

[RFC-2409] Harkins D., Carrel D., "The Internet Key Exchange (IKE)", November 1998

[RFC-2119] Bradner, S., "Key words for use in RFCs to indicate

Requirement Levels", March 1997

[RFC-2407] Piper D., "The Internet IP Security Domain of Interpretation

for ISAKMP", November 1998

8. Authors' Addresses

Tero Kivinen
SSH Communications Security Corp
Fredrikinkatu 42
FIN-00100 HELSINKI
Finland
E-mail: kivinen@ssh.fi