3GPP TSG SA WG3 Security — S3#20 **S3-010604**

27 - 30 November, 2001

Sophia Antipolis, France

Agenda Item:

Source: Ericsson, Nokia, Nortel Networks

Title: Security Mechanism Agreement for SIP Connections

Document for: Informative

1 Scope and objectives

Ericsson, Nokia and Nortel Networks has submitted a new Internet Draft, “Security Mechanism Agreement for SIP Connections” (draft-arkko-sip-sec-agree-00.txt) to IETF.

The purpose of this draft is to (a) correct vulnerability in HTTP Digest authentication for man-in-the-middle attackers, and (b) to allow SIP peers to securely pick the security method they are going to use. A number of proposals have been made that could also be used for 'negotiation' of different SIP parameters, but this proposal attempts to provide security against man-in-the-middle attackers.

The mechanism works both for the hop-by-hop and end-to-end cases, it can be used for negotiating security mechanisms at different protocol layers (as long as they are under the control of the SIP node), and it enables the delivery of different security parameters.

As such, the proposed mechanism should satisfy the 3GPP IMS requirements for security mode set-up. SA3 feedback and analysis on this issue is requested.

The draft will be discussed in the next IETF meeting (Salt Lake City, December 9-14, 2001).

Network Working Group
INTERNET-DRAFT
<draft-arkko-sip-sec-agree-00.txt>
14 November 2001

Jari Arkko
Vesa Torvinen
Ericsson
Tao Haukka
Nokia
Sanjoy Sen
Lee Valerius
Nortel Networks

Security Mechanism Agreement for SIP Connections

1. Status of this Memo

2. Abstract

SIP has a number of security mechanisms for hop-by-hop and end-to-end protection. Some of the security mechanisms have been built in to the SIP protocol, such as HTTP authentication or secure attachments. In these mechanisms there are even alternative algorithms and parameters. Currently, HTTP authentication is known to be vulnerable to so called Bidding-Down attacks where a Man-In-The-Middle attacker simply modifies messages in a way that leads parties to believe the other side only supports weaker algorithms than they actually do. Also, currently it isn't possible to select which security mechanisms to use over a connection. In particular, even if some mechanisms such as OPTIONS or NEGOTIATE were used to make this selection, the selection would be again vulnerable against the Bidding-Down attack. On small networks configuration and software update methods are sufficient to deal with this type of attacks, but on large networks that evolve over time, the security implications are serious: either you deny connections from large amounts of older equipment, or risk losing all value of new algorithms through attacks that are trivial to the attackers.

3. Contents

4. Introduction

Traditionally, security protocols have included facilities to agree on the used mechanisms, algorithms, and other security parameters. The reason for this is that experience has shown algorithm development uncovers problems in old algorithms and produces new ones. Further¡ more, different algorithms are suitable for different situations. Typ¡ ically, protocols also select other parameters beyond algorithms at the same time.

The purpose of this paper is to study whether similar functionality is necessary in SIP [1]. SIP has some security functionality built-in such as different variants of HTTP authentication [4], secure attach¡ ments such as S/MIME, and can also use underlying security protocols such as IPSec/IKE [2], TLS [3]. Some of the built-in security func¡ tionality has also alternative algorithms and other parameters. While some work within the SIP Working Group has been looking towards reduc¡ ing the number of recommended security solutions (e.g. recommend just one lower layer security protocol), we can not expect to cut down the number of items in the whole list to one. There will still be multiple security solutions in SIP. Furthermore, given that security work around SIP is in its early stages, it is likely that new methods will appear in the future, to complete the methods that exist today.

Chapter 5 shows that without a secure method to choose between secu¡ rity mechanisms and/or their parameters, SIP is vulnerable to certain attacks. As the HTTP authentication RFC [4] points out, authentication and integrity protection using multiple alternative methods and algo¡

rithms is vulnerable to Man-in-the-Middle (MITM) attacks. More seri¡ ously, it is hard to know if a SIP peer entity truly can't perform e.g. auth-int QOP in Digest, TLS, or S/MIME, or if a MITM attack is in progress. In small workstation networks these issues are not very

relevant, but the deployment of hundreds of millions of small devices with little or no possibilities for coordinated security policies, let alone software upgrades makes these issues much worse. This conclusion is supported by the requirements from 3GPP [7].

Chapter 6 outlines some possible solutions to these problems, and Chapter 7 documents our proposed solution.

5. The Problem

SIP has alternative security mechanisms such as HTTP authentication / integrity protection, lower layer security protocol(s), S/MIME. It is likely that their use will continue in the future. SIP security is developing, and is likely to see also new solutions in the future, for example along the introduction of SIP for new network access technolo¡ gies. Future services may also bring with themselves different secu¡ rity requirements and methods.

Deployment of large number of SIP-based consumer devices such as 3GPP terminals requires all network devices to be able to accommodate both current and future mechanisms; there is no possiblity for instanta¡ neous change since the new solutions are coming gradually in as new standards and product releases occur. It isn't even possible to upgrade some of the devices without getting completely new hardware.

So, the basic security problem that such a large SIP-based network must consider, how do security mechanisms get selected? It would be desirable to take advantage of new mechanisms as they become available in products.

Firstly, we need to know somehow what security should be applied, and preferably find this out without too many additional roundtrips.

Secondly, selection of security mechanisms MUST be secure. Tradition¡ ally, all security protocols use a secure form of negotiation. For instance, after establishing mutual keys through Diffie-Hellman, IKE sends hashes of the previously sent data -- including the offered crypto mechanisms. This allows the peers to detect if the initial, unprotected offers were tampered with.

The security implications of this are subtle, but do have a fundamen¡ tal importance in building large networks that change over time. Given that the hashes are produced also using algorithms agreed in the first unprotected messages, one could ask what the difference in security

really is. First, assuming hashing is mandatory and only secure algo¡ rithms are used, we still need to prevent MITM attackers from modify¡ ing other parameters, such as whether encryption is provided or not. Secondly, it turns out, however, that there indeed is still a differ¡ ence even for hashes. Let us first assume two peers capable of using both strong and weak security. If the initial offers are not protected in any way, *any* attacker can easily "downgrade" the offers by remov¡ ing the strong options. This would force the two peers to use weak security between them. But if the offers are protected in some way --

such as by hashing, or repeating them later when the selected security is really on -- the situation is different. It would not be sufficient for the attacker to modify a single message. Instead, the attacker would have to modify both the offer message, as well as the message that contains the hash/repetition. More importantly, the attacker would have to forge the weak hash / security that is present in the second message, and would have to do so in real time between the sent offers and the later messages. Otherwise, the peers would notice that the hash is incorrect.

In conclusion, the security difference is making a trivial attack pos¡ sible versus demanding the attacker to break algorithms. An example of where this has a serious consequence is when a network is first deployed with integrity protection (such as an improved HTTP Digest [8, 9]), and then later new devices are added that support also encryption (such as S/MIME). In this situation, an insecure negotia¡ tion procedure allows attackers to trivially force even new devices to use only integrity protection.

It can be asked why the devices would be allowing both weak and strong security in the first place. The answer lies in understanding how net¡ works are deployed, and in the logistical and economical problems in upgrading global networks instantanously. These issues are of particu¡ larly high relevance for networks with a large number of devices, such as the third generation mobile networks. Once millions or even hun¡ dreds of millions of devices have been sold to customers, it becomes impossible to replace them with new devices. Therefore, network equip¡ ment such as SIP proxies must continue to accept even the older equipement that are less capable in terms of security. Similarly, clients wishing to stay in contact regardless of who they call or where they are, have a need to allow both weaker and stronger mecha¡ nisms.

Therefore, we feel that in large networks it is necessary to include some security agreement mechanisms in SIP.

6. Alternative Solutions

Basic SIP features such as OPTIONS and Require, Supported headers are capable of informing peers about various capabilities including secu¡

rity mechanisms. However, the straightforward use of these features does not guarantee a secured agreement.

HTTP Digest algorithm lists [4] are not secure for picking among the digest integrity algorithms, as is described in the RFC itself. More seriously, they have no provisions for allowing encryption to be negotiated. Hence, it would be hard to turn on possible future encryption schemes in a secure manner.

The SIP NEGOTIATE method [5] allows powerful negotiation of various kinds of parameters, including security mechanisms and algorithms. However, it does not allow for secure negotiation as is described in the Internet Draft itself.

The SIP Security Framework [6] also allows for the agreement about the used security mechanisms. However, it does not do this in a secure manner.

7. Proposed Solution

In our opinion, the optimal solution to the SIP security negotiation problem has the following properties:

(a) It allows the selection of security mechanisms, such as lower layer security protocols or secure attachments. It also allows the selection of individual algorithms and parameters where the security functions are integrated in SIP (such as in the case of HTTP authentication or secure attachments).

(b) It allows both end-to-end and hop-by-hop negotiation.

(c) It is secure, i.e. prevents bidding down attacks.

(d) It is capable of running without additional roundtrips. This is important in the cellular environment, where an additional roundtrip could cost 1000 to 1500 ms for the call set up delay.

7.1. Design

We propose a scheme where security features are represented as regular option tags in SIP. If there will ever be any features that require parameters such as key lengths, the option tags can be associated with an optional value field. The client announces a list of supported option tags in its first message, and the server returns its selection in the second message.

In order to secure the agreement, we simply repeat the client's original list of option tags in the clients's first protected request. The server can then proceed to verify that the list has not been modified.

If a modification is detected, the server returns on error or discon¡ nects. The server MUST send a positive answer if and only if the list was not modified.

If the server's selection was changed in transit, the message protec¡ tion fails given that wrong algorithms are being tried to be used. The client's first protected request can be a real request such as INVITE, as the server MUST check the correctness of the lists before it pro¡ ceeds to execute the requested operation.

If the above was enough, we could use the regular SIP Supported header for this purpose. However, in order to be able to support hop-by-hop as well as end-to-end agreement in a controlled fashion (and without a large increase of roundtrips), we need to specify the senders and receivers of the security information. For this purpose we use a method similar to the SIP Security Framework proposal [6].

In the protocol design a trade-off has been made between minimizing roundtrips and making the server stateless. In order to implement the

checking functionality, SIP servers MUST store the state from the pre¡ vious messages. The addition of a single roundtrip would have enabled stateless operation. However, it should be noted that where this method is applied, there are already security associations being cre¡ ated so the SIP nodes are already statefull.

7.2. Header descriptions

The following descriptions are of preliminary nature, and could be syntactically represented in different ways, such as with separate headers.

The Security-Method header indicates who wants security towards whom, and what kind of security. The syntax of this header is as follows:

```
"Security-Method:" to-uri "," from-uri "," mechlist

Where

  to-uri = uri
  from-uri = uri
  mechlist = mechopts *( ";" mechopts )
  mechopts = mechtag *( "," mechtag )
  mechtag = option-tag ["=" token *( ":" token )]
```

The meaning of these fields is as follows:

- The "to-uri" indicates the desired receiver of the information. The value of this field should be a SIP URI. When sent by a client, the value would typically (but not necessarily) contain just the host and

port number parts. The special value "*" signifies all SIP entities along the path.

- The "from-uri" indicates the sender of the security agreement infor¡ mation. The value of this is also a SIP URI. When sent by a client, the value would typically (but not necessarily) include a username part. The special value "*" signifies all SIP entities along the path.

- The "mechlist" represents a list of security mechanisms, all of which must be supported simultaneously on the same connection (such as both HTTP Digest *and* IPsec/IKE).

- The "mechopts" represents a list of alternative security mechanisms. Inside one "mechlist" entry we can have multiple alternative mecha¡ nisms and algorithms. For instance, the the list "org.iana.sip.digest=md5, org.iana.sip.digest=sha1; org.iana.sip.ike" would represent the requirement that one must run simultaneous IPsec/IKE and HTTP Digest with either MD5 or SHA1 inside.

The "mechtag" represent one individual mechanism. The "option-tag" syntax is used for these in order to facilitiate the easy addition of new mechanisms. All option tags starting with "org.iana.sip." MUST be documented in Internet Drafts or RFCs. The initial list of standard¡ ized option-tags is presented below:


```
  org.iana.sip.ike: IPsec/IKE
  org.iana.sip.tls: TLS
  org.iana.sip.digest: HTTP Digest authentication, the algorithm
                       and QoP being optional parameters
  org.iana.sip.smime: S/MIME
```

The optional "token" parameters associated with an "option-tag" can be used to assign parameter values to certain options. This may be use¡ ful to select algorithms, key lengths, or other similar parameters in mechanisms integrated to SIP.

Multiple instances of the same header field can appear in SIP mes¡ sages. Typically, the client inserts its own Security-Method header when it sends a request, and the server/proxy adds its own response. The parameters are in all cases set in an appropriate manner to indi¡ cate in the "to-uri" paremeter the party who inserted the header.

8. Examples

8.1. Selecting Between New and Old Mechanisms

In this example we demonstrate the use of the framework for securing the first hop using some security mechanism, without knowing before¡ hand which methods the server supports. We assume that the client is not willing to reveal any information on what it intends to do, so it

uses OPTIONS in the first message that is sent in the clear. The exam¡ ple starts by a client sending a message to the server, indicating that it is of the new variant that supports both HTTP Digest and TLS in Step 1. In Step 2, the server responds that with its selection and the peers start the security services at Step 3. In Step 4, the client resends its Security-Method header, which the server verifies, and responds with 200 OK.

1. Client -> Server:

   ```
   OPTIONS server SIP/2.0
   Security-Method: sip:client sip:server org.iana.sip.tls,
                    org.iana.sip.digest
   ```

2. Server -> Client:

   ```
   200 OK
   Security-Method: sip:server sip:client org.iana.sip.tls
   ```

3. Security handshake at a lower layer

4. Client -> Server:

   ```
   INVITE server SIP/2.0
   Security-Method: sip:client sip:server org.iana.sip.tls,
                    org.iana.sip.digest
   ```

5. Server -> Client:


```
200 OK
```

In the example we have omitted the returned values of Security-Method in replies for clarity. Typically in SIP the servers do not remove header fields as they answer, they only add new headers.

If this example was run without Security-Method in Step 1, the peers would not know what kind of security the other one supports, and would be forced to error-prone trials.

More seriously, if the Security-Method was omitted in Step 4, the whole process would be prone for MITM attacks. An attacker could spoof "ICMP Port Unreachable" message on the trials, or remove the stronger security option from the header in Step 1, therefore substantially reducing the security.

8.2. Selecting Improved Digest Algorithms

This example attempts to show that the 3GPP requirements on being able to use lightweight security methods over the cellular interface and

secure agreement on algorithms in these methods can be achieved using our method.

In 3GPP networks, the clients make REGISTER operation in their first message, in order to inform the home network that they are at a par¡ ticular location. Due to the properties of 3GPP radio interfaces, it is necessary to optimize the number of roundtrips needed in the whole process. Therefore, we try to parallelize the tasks. It should be noted that the same functionality could be achieved using additional OPTIONS messages. We assume that 3GPP uses an improved form of HTTP Digest authentication, perhaps in the form outlined in [8] or [9] to protect signaling in the first hop. (IPsec AH would also be possible without IKE.) We assume this improved method is called integrity pro¡ tection and denoted by org.iana.sip.integrity.

The example starts by a new version client coming to a new area and learning the address of the local proxy. The client also knows its home server address. We assume that some trust has already been estab¡ lished between the client and the home, and between the client and the proxy. Perhaps this trust is in the form of the nodes belonging under the same PKI, or having distributed shared secrets beforehand.

In Step 1 the client sends a message to the server, indicating that it is of the new variant that supports algorithms MD5 and SHA1 for Digest for the protection of the first hop. The messages are passed onwards to the server through the proxy. In Step 2, the proxy responds that with its selection as well as some end-to-end authentication headers that takes place simultaneously. In Step 3, the integrity protection is turned on and the client sends the next round of REGISTER messages to the server. This includes the repetition of the original security capabilities of the client. In Step 4, the server verifies this list, and responds with 200 OK.

1. Client -> Proxy:


```
REGISTER server SIP/2.0
Security-Method: sip:client sip:proxy org.iana.sip.integrity=md5,
                 org.iana.sip.integrity=sha1
```

2. Proxy -> Client:

```
401 Authentication Required
(Some end-to-end authentication headers)
Security-Method: sip:proxy sip:client org.iana.sip.integrity=md5,
```

3. Client -> Proxy:

```
REGISTER server SIP/2.0
(Some end-to-end authentication headers)
(Some proxy integrity header in SIP)
```

```
   Security-Method: sip:client sip:proxy org.iana.sip.integrity=md5,
                    org.iana.sip.integrity=sha1
```

4. Proxy -> Client:

```
   200 OK
   (Some proxy integrity header in SIP)
```

As in the previous example, if this was run without Security-Method in Step 1, the peers would not know what kind of algorithms the peers support.

Also as in the previous example, removing the repetition of the Secu¡ rity-Method header in Step 3 would open the system to MITM attacks.

8.3. Ensuring Protection on All Hops

In this example the client wishes to verify that the whole path is end-to-end protected with IPsec/IKE. In our example we assume one proxy between the client and the server. The client starts by indicat¡ ing it wants some security all the way, as well as some security on its known hop. Further hops have to take in account the first require¡ ment.

```
Client -> Proxy:
  OPTIONS server SIP/2.0
  Security-Method: * * org.iana.sip.ike
  Security-Method: sip:client sip:proxy org.iana.sip.ike

Proxy -> Server:
  OPTIONS server SIP/2.0
  Security-Method: * * org.iana.sip.ike
  Security-Method: sip:client sip:proxy org.iana.sip.ike
  Security-Method: sip:proxy sip:server org.iana.sip.ike

Server -> Proxy:
  200 OK
  Security-Method: sip:server sip:proxy org.iana.sip.ike
```



```
Proxy -> Client:
  200 OK
  Security-Method: sip:server sip:proxy org.iana.sip.ike
  Security-Method: sip:proxy sip:client org.iana.sip.ike

(Security handshakes at lower layer on both connections)

Client -> Proxy:
  INVITE server SIP/2.0
  Security-Method: * * org.iana.sip.ike
```

```
    Security-Method: sip:client sip:proxy org.iana.sip.ike

  Proxy -> Server:
    OPTIONS server SIP/2.0
    Security-Method: * * org.iana.sip.ike
    Security-Method: sip:client sip:proxy org.iana.sip.ike
    Security-Method: sip:proxy sip:server org.iana.sip.ike

  Server -> Proxy:
    200 OK

  Proxy -> Client:
    200 OK
```

In this example, the number of requirements for security put forward by the Security-Method header increase as the messages travel through the proxy chain. Each hop has to take in account the "*" statements, and act accordingly. After the security is established on all hops, the repeated statements travel through the same path and the proxy and the server verify that the capability lists are the same.

9. Security Considerations

This draft is about making it possible to select between various SIP security mechanisms in a secure manner. In particular, the method pre¡ sented here allow current networks using hop-by-hop mechanisms to later securely upgrade to end-to-end mechanisms without requiring a simultaneous modification in all equipment. Also, the presented method allows SIP entities to request that the complete path through several proxies is protected with lower-layer mechanisms such as TLS. Cur¡ rently this isn't possible.

The method presented in this draft is secure only if the weakest pro¡ posed mechanism offers at least integrity protection. Therefore, we recommend that HTTP Basic authentication SHOULD NOT be used in con¡ junction with this method. We also recommend that HTTP Digest authen¡ tication be upgraded to support the integrity protection of larger parts of SIP messages than it currently does [8, 9].

10. Conclusions

The presented methods appear to correct a known security hole in HTTP Authentication, and in selecting between different security mecha¡ nisms. This is important for deployments in large networks. The

authors seek comments on the proposed approach, and encourage security analysis of both current SIP and the proposal.

11. Acknowledgments

The authors wish to thank Rolf Blom, Hugh Shieh, Gunther Horn, Krister Boman, David Castellanos-Zamora, Aki Niemi, Valtteri Niemi, and mem¡ bers of the 3GPP SA3 group for interesting discussions in this problem space.

12. References

[1] Handley, M., Schulzrinne, H, Schooler, E. and Rosenberg, J., "SIP: Session Initiation Protocol", Work In Progress, draft-ietf-sip-rfc2543bis-03.txt, IETF, May 2001.

[2] S. Kent, R. Atkinson, "Security Architecture for the Internet Pro¡ tocol", RFC 2401, November 1998.

[3] T. Dierks, C. Allen, "The TLS Protocol Version 1.0", RFC 2246, January 1999.

[4] Franks, J. et al, "HTTP Authentication: Basic and Digest Access Authentication", RFC 2617, June 1999.

[5] S. Parameswar, B. Stucker, "The SIP Negotiate Method", Work In Progress, draft-spbs-sip-negotiate-00.txt, IETF, August 2001.

[6] M. Thomas, "SIP Security Framework", draft-thomas-sip-sec-frame¡ work-00.txt. Work In Progress, IETF, July 2001.

[7] M. Garcia, D. Mills, G. Bajko, G. Mayer, F. Derome, H. Shieh, A. Allen, S. Chotai, K. Drage, J. Bharatia, "3GPP requirements on SIP", draft-garcia-sipping-3gpp-reqs-00.txt. Work In Progress, IETF, October 2001.

[8] J. Undery, "SIP Authentication: SIP Digest Access Authentication", draft-undery-sip-digest-00.txt. Work In Progress, IETF, July 2001.

[9] S. Sen, J. Valerius, "Single Hop Message Authentication in SIP", draft-sen-sipping-onehop-digest-00.txt. Work In Progress, IETF, Novem¡ ber 2001.

13. Author's Address

Jari Arkko, Vesa Torvinen
Ericsson
02420 Jorvas
Finland
EMail: Jari.Arkko@ericsson.com, Vesa.Torvinen@ericsson.fi

Tao Haukka
Nokia
Finland

EMail: Tao.Haukka@nokia.com

Sanjoy Sen
Nortel Networks
2735-B Glenville Drive
Richardson, TX 75082, USA
EMail: sanjoy@nortelnetworks.com

Lee Valerius
Nortel Networks
2201 Lakeside Blvd
Richards, TX 75082, USA
EMail: valerius@nortelnetworks.com