Volume : 3 | Issue : 5 | June 2013 ISSN - 2250-1991

Research Paper Engineering

User Authentication Using GSS-API in Session Initiation Protocol

* Mayank Patel ** Darshan Chauhan

* Department of Computer Engineering, L. D. College of Engineering Ahmedabad, India

** Department of Computer Science Engineering, Jodhpur National University Jodhpur, India

ABSTRACT

The Session Initiation Protocol (SIP), which is becoming the de facto standard for the next-generation VoIP networks, is currently receiving much attention in many aspects. One aspect that was not deeply addressed in the original SIP is its authentication procedure. The mandatory and most deployed authentication method used in the Session Initiation Protocol, the Digest access Authentication method, is weak. Other, more secure authentication methods have emerged, but have seen little adoption yet. In this paper, support for using a generic authentication method, the Generic Security Services API, is added to the Session Initiation Protocol. When using this method, the Session Initiation Protocol does not need to support nor implement other authentication methods, only use the provided API library. This enables the Session Initiation Protocol to transparently support and use more secure authentication methods in a unified and generic way.

Keywords : VoIP, SIP, authentication, GSS-API.

I. INTRODUCTION

VoIP (Voice over Internet Protocol) is an interesting technology in the area of telecommunication, because with it voice calls can be delivered using Internet Protocol (IP) networks, the same as is used in transporting IP data. It is also promising since using Internet as a transport channel is more efficient than maintaining a separate telephone network for calls and another for data communication. In VoIP voice is converted to data packets in contrast to PSTN, which realizes voice delivery in circuits witched mode. VoIP packets are, as in any Internet Protocol based system, exposed to loss, delay or bandwidth limitations. When these network restrictions and issues are solved at acceptable level, VoIP technologies can be potential replacement for PSTN. As VoIP is getting popular and usage grows, it is important to consider the security issues on using it. There are many protocols used in VoIP signaling, but Session Initiation Protocol (SIP) [1] is one of the widely used ones. SIP is a signaling protocol functioning at application layer, which can be used to initiate and terminate Voice over IP (VoIP) and multimedia sessions between user clients. [1, 2] Today SIP is the most promising Internet telephone signaling protocol. This paper adds support for the Generic Security Services Application Program Interface (GSS-API) to SIP. Different security requirements may require different authentication mechanisms. Instead of adding support for many different authentication mechanisms in SIP, support for GSS-API will provide a generic interface that makes different authentication methods transparent to the SIP protocol. To negotiate the best available authentication service between two peers, the Simple and Protected GSSAPI NEGOtiation (SPNEGO) mechanism is used on top of GSS-API.

II. Authentication in SIP

In SIP specification [1], the authentication mechanism proposed is HTTP digest based authentication. In SIP terms, HTTP digest mechanism is called the SIP authentication. Originally, HTTP digest is a challenge-response protocol, in which a nonce value is used in challenging the target. The response includes then a checksum of the username, password, nonce value, HTTP method and requested URI. [8] SIP applies the digest mechanism for authenticating users to users or users to proxies, not proxies to proxies. The security between proxies relies on other mechanisms, for example TLS or IPsec. Unfortunately, the DAA is considered weak and is vulnerable to a series of attacks [8], including registration hijacking [4]. A more secure authentication method can be achieved by using Secure MIME (S/MIME). The Secure Multipart Internet Mail Extensions (S/MIME) in SIP [1] is used to carry replicates of SIP header fields inside a MIME body [7]. This enables authentication by the means of signing the replicated header fields to verify the identity of the sender. When the S/MIME header is received, the receiver checks whether the sender's certificate is signed by a trusted authority. A client must support multiple root certificates since there is no consolidated root authority that is trusted by all clients. This and other certificate handling issues like revoking and renewing complicates the use of certificates. Industry support for S/MIME has been limited [8]. Two other authentication methods have emerged within the Internet Engineering Task Force (IETF):

1. The Asserted Identity is intended to work within a trusted environment. An additional, unprotected SIP header is sent in clear that informs that the identity of the client has been checked. Since the SIP header is sent in clear rather than protected by cryptography methods, it can easily be removed by an attacker without any of the communicating peers noticing this.
2. The SIP Strong Identity introduces a new SIP service, the "authentication service", which signs a hash over selected SIP header values, and includes the signature as a SIP header along with a URI that points to the sender's certificate. The receiver computes the same hash and compares the results.

III. GSS-API WITH SPNEGO

The GSS-API [13] provides a generic interface for application layer protocols like SIP, with a layer of abstraction for different security services like authentication, integrity or confidentiality. With the GSS-API, an application does not need to support or implement every authentication method, but use the provided security API [13]. The GSS-API is not a communication protocol in itself, but relies on the application to encapsulate, send, and extract data messages called "tokens" between the client and server. The tokens' content are opaque from the viewpoint of the calling application, and contain authentication data, or, once the authentication is complete, portion of data that the client and server want to sign or encrypt. The tokens are passed through the GSS-API to a range of underlying security mechanisms, ranging from secret-key cryptography, like Kerberos [13], to public-key cryptography, like the Simple Public-Key GSS-API Mechanism (SPKM) [13]. For an application, the use of the GSS-API becomes a standard interface to request authentication, integrity, and confidentiality services in a uniform way. However, GSS-API does not provide credentials needed by the underlying security mechanisms. Both server and client must acquire their respective credentials before GSS-API functions are called. To establish peer entity authentication, a security context is initialized and established. After the security context has been established, additional messages can be exchanged, that is integrity and, optionally, confidentially protected. To initiate and manage a security context, the peers use the context-level GSS-API calls. The client calls GSS_Init_sec_context() that produces a "output token" that is passed to the server. The server then calls GSS_Accept_sec_context() with the received token as input. Depending on the underlying security mechanism, additional token exchanges may be required in the course of context establishment. If so, GSS_S_CONTINUE_NEEDED status is set and additional tokens are passed between the client and server until a security context is established, as depicted in Figure 1.

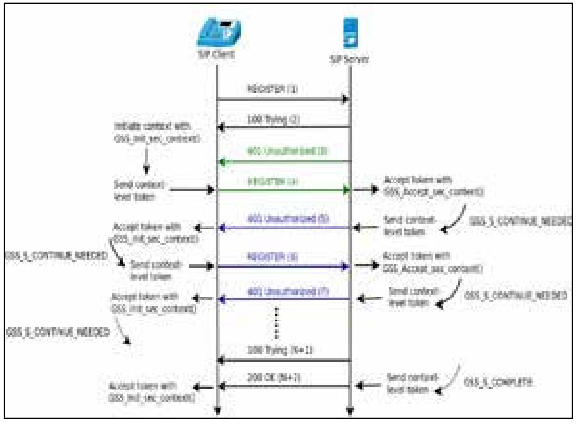


Figure 1: SIP REGISTER message flow with GSS-API security context establishment (authentication).

After a security context has been established, per-message GSS-API calls can be used to protect a message by adding a Message Integrity Code (MIC) with GSS_GetMIC() and verifying the message with GSS_VerifyMIC(). To encrypt and decrypt messages, the peers can use GSS_Wrap() and GSS_Unwrap(). Thus, two different token types exist [13]:

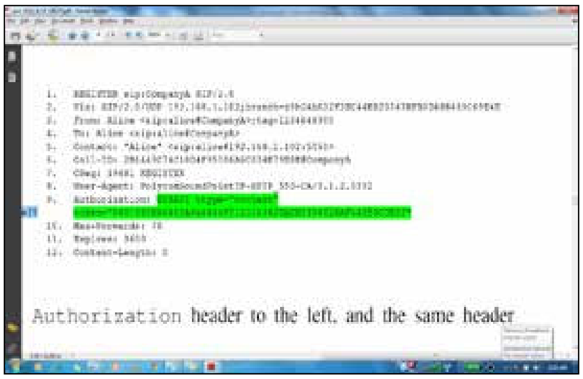
1) Context-level tokens are used when a context is established.
2) Per-message tokens are used after a context has been established, and are used to integrity or confidentiality protect data.

III. Authentication Using GSS-API and SPNEGO IN SIP

Instead of adding numerous different authentication methods to SIP based on different security requirements, it is desirable to keep the changes to the SIP standard to a minimum. Adding support for the GSS-API requires only one small change to the SIP standard, and will open up for a wide range of different authentication methods. In the following subsections, we outline how to include support for the GSS-API into the SIP authentication to replace the original weak Digest Authentication.

A. SIP authentication using GSS-API and SPNEGO

We reuse Digest Authentication headers for GSS-API support, and instead of encapsulate DAA data, we send the GSS-API tokens. An example of new Authorization header with GSS-API data is depicted in Figure 2.



```
REGISTER sip:CompanyA SIP/2.0
Via: SIP/2.0/UDP 192.168.1.102;branch=[illegible]
From: Alice <sip:alice@CompanyA>;tag=[illegible]
To: Alice <sip:alice@CompanyA>
Contact: "Alice" <sip:alice@192.168.1.102:5060>
Call-ID: [illegible]@CompanyA
CSeq: 1468, REGISTER
User-Agent: PolycomSoundPointIP-SPIP_550-UA/3.1.2.0392
Authorization: GSSAPI ttype="context"
    token="[illegible]"
Max-Forwards: 70
Expires: 3600
Content-Length: 0
```

Authorization header to the left, and the same header

Figure 2: Authentication header carrying GSS-API data.

During the initialization of a security context it is necessary to identify the underlying security mechanism to be used. The caller initiating the context indicates at the start of the token the security (authentication) mechanism to be used. The security mechanism is denoted by a unique Object Identifier (OID). However, there is no way for the initiating peer to know which security mechanism the receiving peer supports. If an unsupported "mech type" is requested, the authentication fails. The GSS-API standard resolves this by recommending to manually standardizing on a fixed "mech type" within a domain. Since SIP addresses are designed to be global [28], and not confined to a local domain, a GSS-API negotiation mechanism is required. The SPNEGO is such a GSS-API negotiation mechanism. When using GSS-API with the SPNEGO mechanism, the number of SIP messages between client and server during authentication needs to be increased. During a DAA authentication, the client sends a REGISTER message to the server. The server, upon receiving a REGISTER, challenges the client with a nonce. The client then generates a digest response, a hash value computed over several SIP header values, the nonce, and a shared secret. The client then re-sends the REGISTER message with the digest response embedded. The message flow of a SIP DAA handshake is shown in the first four messages depicted in Figure 1.

In the following paragraphs, the numbers in parentheses refer to the numbers in Figure 1 [13]. When a client comes online and registers itself to a "location service" (SIP server), it does so by sending a SIP REGISTER message (1). We define the token type in the variable ttype. In the following messages, the ttype is set to "context" indicating that these tokens are context-level tokens. The first message (1) does not contain any Authorization header. The server responds with an empty WWW-Authenticate header (3):

REGISTER SIP/2.0
WWW-Authenticate: GSSAPI ttype="context"
token=""

The client then calls GSS_Init_sec_context() with SPNEGO as underlying GSS-API mechanism to negotiate a common authentication mechanism (4). The GSS-API "mech type" is set to SPNEGOs OID 1.3.6.1.5.5.2. The token data might be in binary format, depending on the security mechanism used. Since the SIP headers are in ASCII string format, the token data is base64 encoded:

SIP/2.0 401 Unauthorized
Authorization: GSSAPI ttype="context"
token="0401000B06092A864886F7120..."

The server retrieves the GSS-API data, the token, and passes this to the SPNEGO GSS-API mechanism. In this first initial token, the client embeds authentication data for its first preferred authentication mechanism. This way, should the server accept the clients preferred mechanism, we avoid an extra SIP message round trip. If the client's preferred method was accepted by the server, the server passes the relevant authentication data to the selected authentication mechanism in a 401 SIP message (5). The selected authentication method continues to pass tokens between client and server as many times as necessary to complete the authentication (6-7-N) and establish a security context. Once the security context is established, it sends a 200 OK SIP message (N+2). Should the server have some last GSS-API data to be communicated to the client to complete the security context, it can be carried in a WWW-Authenticate header embedded in the 200 OK message:

```
SIP/2.0 200 OK
WWW-Authenticate: GSSAPI ttype="context"
token="dd02c7c2232759874e1c20558701..."
```

If the client's preferred mechanism is not the server's most preferred mechanism, the server outputs a negotiation token and sends it to the client embedded in a new 401 SIP message (5). The client processes the received SIP message and passes the authentication data to the correct authentication mechanism. The GSS-API then continues as described in the previous paragraph.

IV. Conclusion and future work

Since the only mandatory and widely deployed Digest Access Authentication method in SIP is weak, other more secure authentication methods are desired. In this paper, we have added support for GSS-API in SIP, as well as for the SPNEGO mechanism that is used to negotiate the preferred GSS-API security mechanism supported by both client and server. The required change to the SIP protocol has been kept to a minimum, and the authentication header from DAA has been reused to prevent adding additional SIP headers to the standard. Different VoIP installations have different security requirements that may require different security services. We have shown that the use of the GSS-API provides SIP with a wide range of different authentication methods in a uniform and standardized way. Different authentication methods can be used depending on the different security requirements for each SIP installation. This adds to the flexibility of SIP, like adding a new authentication method, without requiring further changes to the SIP standard.

REFERENCES

[1]. Jan 2011H. Sinnreich and A. B. Johnston, Internet communications using SIP: Delivering VoIP and multimedia services with Session Initiation Protocol, 2nd ed. New York, NY, USA: John Wiley & Sons, Inc., August 2006. | [2]. VoIPSA, "VoIP security and privacy threat taxonomy," Public Realease 1.0, Oct. 2005. [Online]. Available: http://voipsa.org/Activities/VOIPSA Threat Taxonomy 0.1.pdf 1. Nov 2011 | [3]. A. D. Keromytis, "Voice over IP: Risks, Threats and Vulnerabilities," in Proceedings of the Cyber Infrastructure Protection (CIP) Conference, New York, June 2009.[8] H. Dwivedi, Hacking VoIP: Protocols, Attacks, and Countermeasures, 1st ed. No Starch Press, Mar. 2009. | [4]. A. D. Keromytis, "Voice-over-IP security: Research and practice," IEEE Security & Privacy Magazine, vol. 8, no. 2, pp. 76–78, 2010. | [5]. "Research project: EUX2010SEC – Enterprise Unified Exchange Security". | [6]. L. Strand, "VoIP lab as a research tool in the EUX2010SEC project," Norwegian Computing Center, Department of Applied Research in Information Technology, Tech. Rep. DART/08/10, April 2010. | [7]. A. D. Keromytis, "A Survey of Voice Over IP Security Research," in Proceeding of the 5th International Conference on Information Systems Security (ICISS), December 2009, pp. 1 – 17. | [8]. D. Sisalem, J. Floroiu, J. Kuthan, U. Abend, and H. Schulzrinne, SIP Security. WileyBlackwell, Mar. 2009. | [9]. J. Peterson and C. Jennings, "Enhancements for Authenticated Identity Management in the Session Initiation Protocol (SIP)," RFC 4474 (Proposed Standard), Internet Engineering Task Force, Aug. 2006. | [10]. J. Linn, "Generic Security Service Application Program Interface Version 2, Update 1," RFC 2743 (Proposed Standard), Internet Engineering Task Force, Jan. 2000, updated by RFC 5554. [Online] | [11]. D. Todorov, Mechanics of User Identification and Authentication: Fundamentals of Identity Management, 1st ed. Auerbach Publication, Jun. 2007. | [12]. L. Zhu, K. Jaganathan, and S. Hartman, "The Kerberos Version 5 Generic Security Service Application Program Interface (GSSAPI) Mechanism: Version 2," RFC 4121 (Proposed Standard), Internet Engineering Task Force, Jul. 2005. | [13]. L. Strand and W. Leister, Generic Security Services API authentication support for the Session Initiation Protocol, Xpert Publishing Services, Jan 2011, |