Network Working Group
Request for Comments: 4474
Category: Standards Track

J. Peterson
NeuStar
C. Jennings
Cisco Systems

August 2006

Enhancements for Authenticated Identity Management in the Session Initiation Protocol (SIP)

Status of This Memo

Copyright Notice

Abstract

The existing security mechanisms in the Session Initiation Protocol (SIP) are inadequate for cryptographically assuring the identity of the end users that originate SIP requests, especially in an interdomain context. This document defines a mechanism for securely identifying originators of SIP messages. It does so by defining two new SIP header fields, Identity, for conveying a signature used for validating the identity, and Identity-Info, for conveying a reference to the certificate of the signer.

Table of Contents

1. Introduction

This document provides enhancements to the existing mechanisms for authenticated identity management in the Session Initiation Protocol (SIP, RFC 3261 [1]). An identity, for the purposes of this document, is defined as a SIP URI, commonly a canonical address-of-record (AoR) employed to reach a user (such as 'sip:alice@atlanta.example.com').

RFC 3261 stipulates several places within a SIP request where a user can express an identity for themselves, notably the user-populated From header field. However, the recipient of a SIP request has no way to verify that the From header field has been populated appropriately, in the absence of some sort of cryptographic authentication mechanism.

RFC 3261 specifies a number of security mechanisms that can be employed by SIP user agents (UAs), including Digest, Transport Layer Security (TLS), and S/MIME (implementations may support other security schemes as well). However, few SIP user agents today support the end-user certificates necessary to authenticate themselves (via S/MIME, for example), and furthermore Digest authentication is limited by the fact that the originator and destination must share a prearranged secret. It is desirable for SIP user agents to be able to send requests to destinations with which they have no previous association -- just as in the telephone network today, one can receive a call from someone with whom one has no previous association, and still have a reasonable assurance that the person's displayed Caller-ID is accurate. A cryptographic approach, like the one described in this document, can probably provide a much stronger and less-spoofable assurance of identity than the telephone network provides today.

2. Terminology

In this document, the key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" are to be interpreted as described in RFC 2119 [2] and indicate requirement levels for compliant SIP implementations.

3. Background

The usage of many SIP applications and services is governed by authorization policies. These policies may be automated, or they may be applied manually by humans. An example of the latter would be an Internet telephone application that displays the Caller-ID of a caller, which a human may review before answering a call. An example of the former would be a presence service that compares the identity

of potential subscribers to a whitelist before determining whether it should accept or reject the subscription. In both of these cases, attackers might attempt to circumvent these authorization policies through impersonation. Since the primary identifier of the sender of a SIP request, the From header field, can be populated arbitrarily by the controller of a user agent, impersonation is very simple today. The mechanism described in this document aspires to provide a strong identity system for SIP in which authorization policies cannot be circumvented by impersonation.

All RFC 3261-compliant user agents support Digest authentication, which utilizes a shared secret, as a means for authenticating themselves to a SIP registrar. Registration allows a user agent to express that it is an appropriate entity to which requests should be sent for a particular SIP AoR URI (e.g., 'sip:alice@atlanta.example.com').

By the definition of identity used in this document, registration is a proof of the identity of the user to a registrar. However, the credentials with which a user agent proves its identity to a registrar cannot be validated by just any user agent or proxy server -- these credentials are only shared between the user agent and their domain administrator. So this shared secret does not immediately help a user to authenticate to a wide range of recipients. Recipients require a means of determining whether or not the 'return address' identity of a non-REGISTER request (i.e., the From header field value) has legitimately been asserted.

The AoR URI used for registration is also the URI with which a UA commonly populates the From header field of requests in order to provide a 'return address' identity to recipients. From an authorization perspective, if you can prove you are eligible to register in a domain under a particular AoR, you can prove you can legitimately receive requests for that AoR, and accordingly, when you place that AoR in the From header field of a SIP request other than a registration (like an INVITE), you are providing a 'return address' where you can legitimately be reached. In other words, if you are authorized to receive requests for that 'return address', logically, it follows that you are also authorized to assert that 'return address' in your From header field. This is of course only one manner in which a domain might determine how a particular user is authorized to populate the From header field; as an aside, for other sorts of URIs in the From (like anonymous URIs), other authorization policies would apply.

Ideally, then, SIP user agents should have some way of proving to recipients of SIP requests that their local domain has authenticated them and authorized the population of the From header field. This

document proposes a mediated authentication architecture for SIP in which requests are sent to a server in the user's local domain, which authenticates such requests (using the same practices by which the domain would authenticate REGISTER requests). Once a message has been authenticated, the local domain then needs some way to communicate to other SIP entities that the sending user has been authenticated and its use of the From header field has been authorized. This document addresses how that imprimatur of authentication can be shared.

RFC 3261 already describes an architecture very similar to this in Section 26.3.2.2, in which a user agent authenticates itself to a local proxy server, which in turn authenticates itself to a remote proxy server via mutual TLS, creating a two-link chain of transitive authentication between the originator and the remote domain. While this works well in some architectures, there are a few respects in which this is impractical. For one, transitive trust is inherently weaker than an assertion that can be validated end-to-end. It is possible for SIP requests to cross multiple intermediaries in separate administrative domains, in which case transitive trust becomes even less compelling.

One solution to this problem is to use 'trusted' SIP intermediaries that assert an identity for users in the form of a privileged SIP header. A mechanism for doing so (with the P-Asserted-Identity header) is given in [12]. However, this solution allows only hop-by-hop trust between intermediaries, not end-to-end cryptographic authentication, and it assumes a managed network of nodes with strict mutual trust relationships, an assumption that is incompatible with widespread Internet deployment.

Accordingly, this document specifies a means of sharing a cryptographic assurance of end-user SIP identity in an interdomain or intradomain context that is based on the concept of an 'authentication service' and a new SIP header, the Identity header. Note that the scope of this document is limited to providing this identity assurance for SIP requests; solving this problem for SIP responses is more complicated and is a subject for future work.

This specification allows either a user agent or a proxy server to provide identity services and to verify identities. To maximize end-to-end security, it is obviously preferable for end-users to acquire their own certificates and corresponding private keys; if they do, they can act as an authentication service. However, end-user certificates may be neither practical nor affordable, given the difficulties of establishing a Public Key Infrastructure (PKI) that extends to end-users, and moreover, given the potentially large number of SIP user agents (phones, PCs, laptops, PDAs, gaming

devices) that may be employed by a single user. In such environments, synchronizing keying material across multiple devices may be very complex and requires quite a good deal of additional endpoint behavior. Managing several certificates for the various devices is also quite problematic and unpopular with users. Accordingly, in the initial use of this mechanism, it is likely that intermediaries will instantiate the authentication service role.

4. Overview of Operations

This section provides an informative (non-normative) high-level overview of the mechanisms described in this document.

Imagine the case where Alice, who has the home proxy of example.com and the address-of-record sip:alice@example.com, wants to communicate with sip:bob@example.org.

Alice generates an INVITE and places her identity in the From header field of the request. She then sends an INVITE over TLS to an authentication service proxy for her domain.

The authentication service authenticates Alice (possibly by sending a Digest authentication challenge) and validates that she is authorized to assert the identity that is populated in the From header field. This value may be Alice's AoR, or it may be some other value that the policy of the proxy server permits her to use. It then computes a hash over some particular headers, including the From header field and the bodies in the message. This hash is signed with the certificate for the domain (example.com, in Alice's case) and inserted in a new header field in the SIP message, the 'Identity' header.

The proxy, as the holder of the private key of its domain, is asserting that the originator of this request has been authenticated and that she is authorized to claim the identity (the SIP address-of-record) that appears in the From header field. The proxy also inserts a companion header field, Identity-Info, that tells Bob how to acquire its certificate, if he doesn't already have it.

When Bob's domain receives the request, it verifies the signature provided in the Identity header, and thus can validate that the domain indicated by the host portion of the AoR in the From header field authenticated the user, and permitted the user to assert that From header field value. This same validation operation may be performed by Bob's user agent server (UAS).

5. Authentication Service Behavior

This document defines a new role for SIP entities called an authentication service. The authentication service role can be instantiated by a proxy server or a user agent. Any entity that instantiates the authentication service role MUST possess the private key of a domain certificate. Intermediaries that instantiate this role MUST be capable of authenticating one or more SIP users that can register in that domain. Commonly, this role will be instantiated by a proxy server, since these entities are more likely to have a static hostname, hold a corresponding certificate, and have access to SIP registrar capabilities that allow them to authenticate users in their domain. It is also possible that the authentication service role might be instantiated by an entity that acts as a redirect server, but that is left as a topic for future work.

SIP entities that act as an authentication service MUST add a Date header field to SIP requests if one is not already present (see Section 9 for information on how the Date header field assists verifiers). Similarly, authentication services MUST add a Content-Length header field to SIP requests if one is not already present; this can help verifiers to double-check that they are hashing exactly as many bytes of message-body as the authentication service when they verify the message.

Entities instantiating the authentication service role perform the following steps, in order, to generate an Identity header for a SIP request:

Step 1:

The authentication service MUST extract the identity of the sender from the request. The authentication service takes this value from the From header field; this AoR will be referred to here as the 'identity field'. If the identity field contains a SIP or SIP Secure (SIPS) URI, the authentication service MUST extract the hostname portion of the identity field and compare it to the domain(s) for which it is responsible (following the procedures in RFC 3261, Section 16.4, used by a proxy server to determine the domain(s) for which it is responsible). If the identity field uses the TEL URI scheme, the policy of the authentication service determines whether or not it is responsible for this identity; see Section 11 for more information. If the authentication service is not responsible for the identity in question, it SHOULD process and forward the request normally, but it MUST NOT add an Identity header; see below for more information on authentication service handling of an existing Identity header.

Step 2:

The authentication service MUST determine whether or not the sender of the request is authorized to claim the identity given in the identity field. In order to do so, the authentication service MUST authenticate the sender of the message. Some possible ways in which this authentication might be performed include:

> If the authentication service is instantiated by a SIP intermediary (proxy server), it may challenge the request with a 407 response code using the Digest authentication scheme (or viewing a Proxy-Authentication header sent in the request, which was sent in anticipation of a challenge using cached credentials, as described in RFC 3261, Section 22.3). Note that if that proxy server is maintaining a TLS connection with the client over which the client had previously authenticated itself using Digest authentication, the identity value obtained from that previous authentication step can be reused without an additional Digest challenge.
>
> If the authentication service is instantiated by a SIP user agent, a user agent can be said to authenticate its user on the grounds that the user can provision the user agent with the private key of the domain, or preferably by providing a password that unlocks said private key.

Authorization of the use of a particular username in the From header field is a matter of local policy for the authentication service, one that depends greatly on the manner in which authentication is performed. For example, one policy might be as follows: the username given in the 'username' parameter of the Proxy-Authorization header MUST correspond exactly to the username in the From header field of the SIP message. However, there are many cases in which this is too limiting or inappropriate; a realm might use 'username' parameters in Proxy-Authorization that do not correspond to the user-portion of SIP From headers, or a user might manage multiple accounts in the same administrative domain. In this latter case, a domain might maintain a mapping between the values in the 'username' parameter of Proxy-Authorization and a set of one or more SIP URIs that might legitimately be asserted for that 'username'. For example, the username can correspond to the 'private identity' as defined in Third Generation Partnership Project (3GPP), in which case the From header field can contain any one of the public identities associated with this private identity. In this instance, another policy might be as follows: the URI in the From header field MUST correspond exactly to one of the mapped URIs associated with the 'username' given in the Proxy-Authorization header. Various exceptions to such policies might arise for cases like anonymity; if the AoR asserted in the From

header field uses a form like 'sip:anonymous@example.com', then the 'example.com' proxy should authenticate that the user is a valid user in the domain and insert the signature over the From header field as usual.

Note that this check is performed on the addr-spec in the From header field (e.g., the URI of the sender, like 'sip:alice@atlanta.example.com'); it does not convert the display-name portion of the From header field (e.g., 'Alice Atlanta'). Authentication services MAY check and validate the display-name as well, and compare it to a list of acceptable display-names that may be used by the sender; if the display-name does not meet policy constraints, the authentication service MUST return a 403 response code. The reason phrase should indicate the nature of the problem; for example, "Inappropriate Display Name". However, the display-name is not always present, and in many environments the requisite operational procedures for display-name validation may not exist. For more information, see Section 13.2.

Step 3:

The authentication service SHOULD ensure that any preexisting Date header in the request is accurate. Local policy can dictate precisely how accurate the Date must be; a RECOMMENDED maximum discrepancy of ten minutes will ensure that the request is unlikely to upset any verifiers. If the Date header contains a time different by more than ten minutes from the current time noted by the authentication service, the authentication service SHOULD reject the request. This behavior is not mandatory because a user agent client (UAC) could only exploit the Date header in order to cause a request to fail verification; the Identity header is not intended to provide a source of non-repudiation or a perfect record of when messages are processed. Finally, the authentication service MUST verify that the Date header falls within the validity period of its certificate. For more information on the security properties associated with the Date header field value, see Section 9.

Step 4:

The authentication service MUST form the identity signature and add an Identity header to the request containing this signature. After the Identity header has been added to the request, the authentication service MUST also add an Identity-Info header. The Identity-Info header contains a URI from which its certificate can be acquired. Details on the generation of both of these headers are provided in Section 9.

Finally, the authentication service MUST forward the message normally.

5.1. Identity within a Dialog and Retargeting

Retargeting is broadly defined as the alteration of the Request-URI by intermediaries. More specifically, retargeting supplants the original target URI with one that corresponds to a different user, a user that is not authorized to register under the original target URI. By this definition, retargeting does not include translation of the Request-URI to a contact address of an endpoint that has registered under the original target URI, for example.

When a dialog-forming request is retargeted, this can cause a few wrinkles for the Identity mechanism when it is applied to requests sent in the backwards direction within a dialog. This section provides some non-normative considerations related to this case.

When a request is retargeted, it may reach a SIP endpoint whose user is not identified by the URI designated in the To header field value. The value in the To header field of a dialog-forming request is used as the From header field of requests sent in the backwards direction during the dialog, and is accordingly the header that would be signed by an authentication service for requests sent in the backwards direction. In retargeting cases, if the URI in the From header does not identify the sender of the request in the backwards direction, then clearly it would be inappropriate to provide an Identity signature over that From header. As specified above, if the authentication service is not responsible for the domain in the From header field of the request, it MUST NOT add an Identity header to the request, and it should process/forward the request normally.

Any means of anticipating retargeting, and so on, is outside the scope of this document, and likely to have equal applicability to response identity as it does to requests in the backwards direction within a dialog. Consequently, no special guidance is given for implementers here regarding the 'connected party' problem; authentication service behavior is unchanged if retargeting has occurred for a dialog-forming request. Ultimately, the authentication service provides an Identity header for requests in the backwards dialog when the user is authorized to assert the identity given in the From header field, and if they are not, an Identity header is not provided.

For further information on the problems of response identity and the potential solution spaces, see [15].

6. Verifier Behavior

This document introduces a new logical role for SIP entities called a server. When a verifier receives a SIP message containing an Identity header, it may inspect the signature to verify the identity of the sender of the message. Typically, the results of a verification are provided as input to an authorization process that is outside the scope of this document. If an Identity header is not present in a request, and one is required by local policy (for example, based on a per-sending-domain policy, or a per-sending-user policy), then a 428 'Use Identity Header' response MUST be sent.

In order to verify the identity of the sender of a message, an entity acting as a verifier MUST perform the following steps, in the order here specified.

Step 1:

The verifier MUST acquire the certificate for the signing domain. Implementations supporting this specification SHOULD have some means of retaining domain certificates (in accordance with normal practices for certificate lifetimes and revocation) in order to prevent themselves from needlessly downloading the same certificate every time a request from the same domain is received. Certificates cached in this manner should be indexed by the URI given in the Identity-Info header field value.

Provided that the domain certificate used to sign this message is not previously known to the verifier, SIP entities SHOULD discover this certificate by dereferencing the Identity-Info header, unless they have some more efficient implementation-specific way of acquiring certificates for that domain. If the URI scheme in the Identity-Info header cannot be dereferenced, then a 436 'Bad Identity-Info' response MUST be returned. The verifier processes this certificate in the usual ways, including checking that it has not expired, that the chain is valid back to a trusted certification authority (CA), and that it does not appear on revocation lists. Once the certificate is acquired, it MUST be validated following the procedures in RFC 3280 [9]. If the certificate cannot be validated (it is self-signed and untrusted, or signed by an untrusted or unknown certificate authority, expired, or revoked), the verifier MUST send a 437 'Unsupported Certificate' response.

Step 2:

The verifier MUST follow the process described in Section 13.4 to determine if the signer is authoritative for the URI in the From header field.

Step 3:

The verifier MUST verify the signature in the Identity header field, following the procedures for generating the hashed digest-string described in Section 9. If a verifier determines that the signature on the message does not correspond to the reconstructed digest-string, then a 438 'Invalid Identity Header' response MUST be returned.

Step 4:

The verifier MUST validate the Date, Contact, and Call-ID headers in the manner described in Section 13.1; recipients that wish to verify Identity signatures MUST support all of the operations described there. It must furthermore ensure that the value of the Date header falls within the validity period of the certificate whose corresponding private key was used to sign the Identity header.

7. Considerations for User Agent

This mechanism can be applied opportunistically to existing SIP deployments; accordingly, it requires no change to SIP user agent behavior in order for it to be effective. However, because this mechanism does not provide integrity protection between the UAC and the authentication service, a UAC SHOULD implement some means of providing this integrity. TLS would be one such mechanism, which is attractive because it MUST be supported by SIP proxy servers, but is potentially problematic because it is a hop-by-hop mechanism. See Section 13.3 for more information about securing the channel between the UAC and the authentication service.

When a UAC sends a request, it MUST accurately populate the From header field with a value corresponding to an identity that it believes it is authorized to claim. In a request, it MUST set the URI portion of its From header to match a SIP, SIPS, or TEL URI AoR that it is authorized to use in the domain (including anonymous URIs, as described in RFC 3323 [3]). In general, UACs SHOULD NOT use the TEL URI form in the From header field (see Section 11).

Note that this document defines a number of new 4xx response codes. If user agents support these response codes, they will be able to respond intelligently to Identity-based error conditions.

The UAC MUST also be capable of sending requests, including mid-call requests, through an 'outbound' proxy (the authentication service). The best way to accomplish this is using pre-loaded Route headers and loose routing. For a given domain, if an entity that can instantiate the authentication service role is not in the path of dialog-forming

requests, identity for mid-dialog requests in the backwards direction cannot be provided.

As a recipient of a request, a user agent that can verify signed identities should also support an appropriate user interface to render the validity of identity to a user. User agent implementations SHOULD differentiate signed From header field values from unsigned From header field values when rendering to an end-user the identity of the sender of a request.

8. Considerations for Proxy Servers

Domain policy may require proxy servers to inspect and verify the identity provided in SIP requests. A proxy server may wish to ascertain the identity of the sender of the message to provide spam prevention or call control services. Even if a proxy server does not act as an authentication service, it MAY validate the Identity header before it makes a forwarding decision for a request. Proxy servers MUST NOT remove or modify an existing Identity or Identity-Info header in a request.

9. Header Syntax

This document specifies two new SIP headers: Identity and Identity-Info. Each of these headers can appear only once in a SIP message. The grammar for these two headers is (following the ABNF [6] in RFC 3261 [1]):

```
Identity = "Identity" HCOLON signed-identity-digest
signed-identity-digest = LDQUOT 32LHEX RDQUOT

Identity-Info = "Identity-Info" HCOLON ident-info
                 *( SEMI ident-info-params )
ident-info = LAQUOT absoluteURI RAQUOT
ident-info-params = ident-info-alg / ident-info-extension
ident-info-alg = "alg" EQUAL token
ident-info-extension = generic-param
```

The signed-identity-digest is a signed hash of a canonical string generated from certain components of a SIP request. To create the contents of the signed-identity-digest, the following elements of a SIP message MUST be placed in a bit-exact string in the order specified here, separated by a vertical line, "|" or %x7C, character:

- The AoR of the UA sending the message, or addr-spec of the From header field (referred to occasionally here as the 'identity field').

- The addr-spec component of the To header field, which is the AoR to which the request is being sent.
- The callid from Call-Id header field.
- The digit (1*DIGIT) and method (method) portions from CSeq header field, separated by a single space (ABNF SP, or %x20). Note that the CSeq header field allows linear whitespace (LWS) rather than SP to separate the digit and method portions, and thus the CSeq header field may need to be transformed in order to be canonicalized. The authentication service MUST strip leading zeros from the 'digit' portion of the Cseq before generating the digest-string.
- The Date header field, with exactly one space each for each SP and the weekday and month items case set as shown in BNF in RFC 3261. RFC 3261 specifies that the BNF for weekday and month is a choice amongst a set of tokens. The RFC 2234 rules for the BNF specify that tokens are case sensitive. However, when used to construct the canonical string defined here, the first letter of each week and month MUST be capitalized, and the remaining two letters must be lowercase. This matches the capitalization provided in the definition of each token. All requests that use the Identity mechanism MUST contain a Date header.
- The addr-spec component of the Contact header field value. If the request does not contain a Contact header, this field MUST be empty (i.e., there will be no whitespace between the fourth and fifth "|" characters in the canonical string).
- The body content of the message with the bits exactly as they are in the Message (in the ABNF for SIP, the message-body). This includes all components of multipart message bodies. Note that the message-body does NOT include the CRLF separating the SIP headers from the message-body, but does include everything that follows that CRLF. If the message has no body, then message-body will be empty, and the final "|" will not be followed by any additional characters.

For more information on the security properties of these headers, and why their inclusion mitigates replay attacks, see Section 13 and [5]. The precise formulation of this digest-string is, therefore (following the ABNF [6] in RFC 3261 [1]):

```
digest-string = addr-spec "|" addr-spec "|" callid "|"
                1*DIGIT SP Method "|" SIP-date "|" [ addr-spec ] "|"
                message-body
```

Note again that the first addr-spec MUST be taken from the From header field value, the second addr-spec MUST be taken from the To header field value, and the third addr-spec MUST be taken from the Contact header field value, provided the Contact header is present in the request.

After the digest-string is formed, it MUST be hashed and signed with the certificate for the domain. The hashing and signing algorithm is specified by the 'alg' parameter of the Identity-Info header (see below for more information on Identity-Info header parameters). This document defines only one value for the 'alg' parameter: 'rsa-sha1'; further values MUST be defined in a Standards Track RFC, see Section 14.7 for more information. All implementations of this specification MUST support 'rsa-sha1'. When the 'rsa-sha1' algorithm is specified in the 'alg' parameter of Identity-Info, the hash and signature MUST be generated as follows: compute the results of signing this string with sha1WithRSAEncryption as described in RFC 3370 [7] and base64 encode the results as specified in RFC 3548 [8]. A 1024-bit or longer RSA key MUST be used. The result is placed in the Identity header field. For detailed examples of the usage of this algorithm, see Section 10.

The 'absoluteURI' portion of the Identity-Info header MUST contain a URI which dereferences to a resource containing the certificate of the authentication service. All implementations of this specification MUST support the use of HTTP and HTTPS URIs in the Identity-Info header. Such HTTP and HTTPS URIs MUST follow the conventions of RFC 2585 [10], and for those URIs the indicated resource MUST be of the form 'application/pkix-cert' described in that specification. Note that this introduces key lifecycle management concerns; were a domain to change the key available at the Identity-Info URI before a verifier evaluates a request signed by an authentication service, this would cause obvious verifier failures. When a rollover occurs, authentication services SHOULD thus provide new Identity-Info URIs for each new certificate, and SHOULD continue to make older key acquisition URIs available for a duration longer than the plausible lifetime of a SIP message (an hour would most likely suffice).

The Identity-Info header field MUST contain an 'alg' parameter. No other parameters are defined for the Identity-Info header in this document. Future Standards Track RFCs may define additional Identity-Info header parameters.

This document adds the following entries to Table 2 of RFC 3261 [1]:

Header field	where	proxy	ACK	BYE	CAN	INV	OPT	REG
Identity	R	a	o	o	-	o	o	o

SUB	NOT	REF	INF	UPD	PRA
o	o	o	o	o	o

Header field	where	proxy	ACK	BYE	CAN	INV	OPT	REG
Identity-Info	R	a	o	o	-	o	o	o

SUB	NOT	REF	INF	UPD	PRA
o	o	o	o	o	o

Note, in the table above, that this mechanism does not protect the CANCEL method. The CANCEL method cannot be challenged, because it is hop-by-hop, and accordingly authentication service behavior for CANCEL would be significantly limited. Note as well that the REGISTER method uses Contact header fields in very unusual ways that complicate its applicability to this mechanism, and the use of Identity with REGISTER is consequently a subject for future study, although it is left as optional here for forward-compatibility reasons. The Identity and Identity-Info header MUST NOT appear in CANCEL.

10. Compliance Tests and Examples

The examples in this section illustrate the use of the Identity header in the context of a SIP transaction. Implementers are advised to verify their compliance with the specification against the following criteria:

- Implementations of the authentication service role MUST generate identical base64 identity strings to the ones shown in the Identity headers in these examples when presented with the source message and utilizing the appropriate supplied private key for the domain in question.
- Implementations of the verifier role MUST correctly validate the given messages containing the Identity header when utilizing the supplied certificates (with the caveat about self-signed certificates below).

Note that the following examples use self-signed certificates, rather than certificates issued by a recognized certificate authority. The use of self-signed certificates for this mechanism is NOT RECOMMENDED, and it appears here only for illustrative purposes. Therefore, in compliance testing, implementations of verifiers SHOULD generate appropriate warnings about the use of self-signed certificates. Also, the example certificates in this section have placed their domain name subject in the subjectAltName field; in practice, certificate authorities may place domain names in other locations in the certificate (see Section 13.4 for more information).

Note that all examples in this section use the 'rsa-sha1' algorithm.

Bit-exact reference files for these messages and their various transformations are supplied in Appendix B.

10.1. Identity-Info with a Singlepart MIME body

Consider the following private key and certificate pair assigned to 'atlanta.example.com' (rendered in OpenSSL format).

```
-----BEGIN RSA PRIVATE KEY-----
MIICXQIBAAKBgQDPPMBtHVoPkXV+Z6jq1LsgfTELVWpy2BVUffJMPH06LL0cJSQO
aIeVzIojzWtpauB7IylZKlAjB5f429tRuoUiedCwMLKblWAqZt6eHWpCNZJ7lONc
IEwnmh2nAccKk83Lp/VH3tgAS/43DQoX2sndnYh+g8522Pzwg7EGWspzzwIDAQAB
AoGBAK0W3tnEFD7AjVQAnJNXDtx59Aa1Vu2JEXe6oi+OrkFysJjbZJwsLmKtrgtt
PXOU8t2mZpi0wK4hX4tZhntiwGKkUPC3h9Bjp+GerifP341RMyMO+6fPgjqOzUDw
+rPjjMpwD7AkcEcqDgbTrZnWv/QnCSaaF3xkUGfFkLx5OKcRAkEA7UxnsE8XaT30
tP/UUc51gNk2KGKgxQQTHopBcew9yfeCRFhvdL7jpaGatEi5iZwGGQQDVOVHUN1H
0YLpHQjRowJBAN+R2bvA/Nimq464ZgnelEDPqaEAZWaD3kOfhS9+vL7oqES+u5E0
J7kXb7ZkiSVUg9XU/8PxMKx/DAz0dUmOL+UCQH8C9ETUMI2uEbqHbBdVUGNk364C
DFcndSxVh+34KqJdjiYSx6VPPv26X9m7S0OydTkSgs3/4ooPxo8HaMqXm80CQB+r
xbB3UlpOohcBwFK9mTrlMB6Cs9ql66KgwnlL9ukEhHHYozGatdXeoBCyhUsogdSU
6/aSAFcvWEGtj7/vyJECQQCCS1lKgEXoNQPqONalvYhyyMZRXFLdD4gbwRPK1uXK
Ypk3CkfFzOyfjeLcGPxXzq2qzuHzGTDxZ9PAepwX4RSk
-----END RSA PRIVATE KEY-----
-----BEGIN CERTIFICATE-----
MIIC3TCCAkagAwIBAgIBADANBgkqhkiG9w0BAQUFADBZMQswCQYDVQQGEwJVUzEL
MAkGA1UECAwCR0ExEDAOBgNVBAcMB0F0bGFudGExDTALBgNVBAoMBElFVEYxHDAa
BgNVBAMME2F0bGFudGEuZXhhbXBsZS5jb20wHhcNMDUxMDI0MDYzNjA2WhcNMDYx
MDI0MDYzNjA2WjBZMQswCQYDVQQGEwJVUzELMAkGA1UECAwCR0ExEDAOBgNVBAcM
B0F0bGFudGExDTALBgNVBAoMBElFVEYxHDAaBgNVBAMME2F0bGFudGEuZXhhbXBs
ZS5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAM88wG0dWg+RdX5nqOrU
uyB9MQtVanLYFVR98kw8fTosvRwlJA5oh5XMiiPNa2lq4HsjKVkqUCMHl/jb21G6
hSJ50LAwspuVYCpm3p4dakI1knuU41wgTCeaHacBxwqTzcun9Ufe2ABL/jcNChfa
yd2diH6DznbY/PCDsQZaynPPAgMBAAGjgbQwgbEwHQYDVR0OBBYEFNmU/MrbVYcE
KDr/20WISrG1j1rNMIGBBgNVHSMEejB4gBTZlPzK21WHBCg6/9tFiEqxtY9azaFd
pFswWTELMAkGA1UEBhMCVVMxCzAJBgNVBAgMAkdBMRAwDgYDVQQHDAdBdGxhbnRh
```

```
MQ0wCwYDVQQKDARJRVRGMRwwGgYDVQQDDBNhdGxhbnRhLmV4YW1wbGUuY29tggEA
MAwGA1UdEwQFMAMBAf8wDQYJKoZIhvcNAQEFBQADgYEADdQYtswBDmTSTq0mt211
7alm/XGFrb2zdbU0vorxRdOZ04qMyrIpXG1LEmnEOgcocyrXRBvq5p6WbZAcEQk0
DsE3Ve0Nc8x9nmvljW7GsMGFCnCuo4ODTf/1lGdVr9DeCzcj10YUQ3MRemDMXhY2
CtDisLWl7SXOORcZAi1oU9w=
-----END CERTIFICATE-----
```

A user of atlanta.example.com, Alice, wants to send an INVITE to bob@biloxi.example.org. She therefore creates the following INVITE request, which she forwards to the atlanta.example.org proxy server that instantiates the authentication service role:

```
INVITE sip:bob@biloxi.example.org SIP/2.0
Via: SIP/2.0/TLS pc33.atlanta.example.com;branch=z9hG4bKnashds8
To: Bob <sip:bob@biloxi.example.org>
From: Alice <sip:alice@atlanta.example.com>;tag=1928301774
Call-ID: a84b4c76e66710
CSeq: 314159 INVITE
Max-Forwards: 70
Date: Thu, 21 Feb 2002 13:02:03 GMT
Contact: <sip:alice@pc33.atlanta.example.com>
Content-Type: application/sdp
Content-Length: 147

v=0
o=UserA 2890844526 2890844526 IN IP4 pc33.atlanta.example.com
s=Session SDP
c=IN IP4 pc33.atlanta.example.com
t=0 0
m=audio 49172 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

When the authentication service receives the INVITE, it authenticates Alice by sending a 407 response. As a result, Alice adds an Authorization header to her request, and resends to the atlanta.example.com authentication service. Now that the service is sure of Alice's identity, it calculates an Identity header for the request. The canonical string over which the identity signature will be generated is the following (note that the first line wraps because of RFC editorial conventions):

```
sip:alice@atlanta.example.com|sip:bob@biloxi.example.org|
a84b4c76e66710|314159 INVITE|Thu, 21 Feb 2002 13:02:03 GMT|
sip:alice@pc33.atlanta.example.com|v=0
o=UserA 2890844526 2890844526 IN IP4 pc33.atlanta.example.com
s=Session SDP
c=IN IP4 pc33.atlanta.example.com
t=0 0
```

```
m=audio 49172 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

The resulting signature (sha1WithRsaEncryption) using the private RSA key given above, with base64 encoding, is the following:

```
ZYNBbHC00VMZr2kZt6VmCvPonWJMGvQTBDqghoWeLxJfzB2a1pxAr3VgrB0SsSAa
ifsRdiOPoQZYOy2wrVghuhcsMbHWUSFxI6p6q5TOQXHMmz6uEo3svJsSH49thyGn
FVcnyaZ++yRlBYYQTLqWzJ+KVhPKbfU/pryhVn9Yc6U=
```

Accordingly, the atlanta.example.com authentication service will create an Identity header containing that base64 signature string (175 bytes). It will also add an HTTPS URL where its certificate is made available. With those two headers added, the message looks like the following:

```
INVITE sip:bob@biloxi.example.org SIP/2.0
Via: SIP/2.0/TLS pc33.atlanta.example.com;branch=z9hG4bKnashds8
To: Bob <sip:bob@biloxi.example.org>
From: Alice <sip:alice@atlanta.example.com>;tag=1928301774
Call-ID: a84b4c76e66710
CSeq: 314159 INVITE
Max-Forwards: 70
Date: Thu, 21 Feb 2002 13:02:03 GMT
Contact: <sip:alice@pc33.atlanta.example.com>
Identity:
  "ZYNBbHC00VMZr2kZt6VmCvPonWJMGvQTBDqghoWeLxJfzB2a1pxAr3VgrB0SsSAa
   ifsRdiOPoQZYOy2wrVghuhcsMbHWUSFxI6p6q5TOQXHMmz6uEo3svJsSH49thyGn
   FVcnyaZ++yRlBYYQTLqWzJ+KVhPKbfU/pryhVn9Yc6U="
Identity-Info: <https://atlanta.example.com/atlanta.cer>;alg=rsa-sha1
Content-Type: application/sdp
Content-Length: 147

v=0
o=UserA 2890844526 2890844526 IN IP4 pc33.atlanta.example.com
s=Session SDP
c=IN IP4 pc33.atlanta.example.com
t=0 0
m=audio 49172 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

atlanta.example.com then forwards the request normally. When Bob receives the request, if he does not already know the certificate of atlanta.example.com, he dereferences the URL in the Identity-Info header to acquire the certificate. Bob then generates the same canonical string given above, from the same headers of the SIP request. Using this canonical string, the signed digest in the Identity header, and the certificate discovered by dereferencing the

Identity-Info header, Bob can verify that the given set of headers and the message body have not been modified.

10.2. Identity for a Request with No MIME Body or Contact

Consider the following private key and certificate pair assigned to "biloxi.example.org".

```
-----BEGIN RSA PRIVATE KEY-----
MIICXgIBAAKBgQC/obBYLRMPjskrAqWOiGPAUxI3/m2ti7ix4caqCTAuFX5cLegQ
7nmquLOHfIhxVIqT2f06UA0lOo2NVofK9G7MTkVbVNiyAlLYUDEj7XWLDICf3ZHL
6Fr/+CF7wrQ9r4kv7XiJKxodVCCd/DhCT9Gp+VDoe8HymqOW/KsneriyIwIDAQAB
AoGBAJ7fsFIKXKkjWgj8ksGOthS3Sn19xPSCyEdBxfEm2Pj7/Nzzeli/PcOaic0k
JALBcnqN2fHEeIGK/9xUBxTufgQYVJqvyHERs6rXX/iT4Ynm9t1905EiQ9ZpHsrI
/AMMUYA1QrGgAIHvZLVLzq+9KLDEZ+HQbuCLJXF+6bl0Eb5BAkEA636oMANp0Qa3
mYWEQ2utmGsYxkXSfyBb18TCOwCty0ndBR24zyOJF2NbZS98Lz+Ga25hfIGw/JHK
nD9bOE88UwJBANBRSpd4bmS+m48R/13tRESAtHqydNinX0kS/RhwHr7mkHTU3k/M
FxQtx34I3GKzaZxMn0A66KS9v/SHdnF+ePECQQCGe7QshyZ8uitLPtZDclCWhEKH
qAQHmUEZvUF2VHLrbukLLOgHUrHNa24cILv4d3yaCVUetymNcuyTwhKj24wFAkAO
z/jx1EplN3hwL+NsllZoWI58uvu7/Aq2c3czqaVGBbb317sHCYgKk0bAG3kwO3mi
93/LXWT1cdiYVpmBcHDBAkEAmpgkFj+xZu5gWASY5ujv+FCMP0WwaH5hTnXu+tKe
PJ3d2IJZKxGnl6itKRN7GeRh9PSK0kZSqGFeVrvsJ4Nopg==
-----END RSA PRIVATE KEY-----
-----BEGIN CERTIFICATE-----
MIIC1jCCAj+gAwIBAgIBADANBgkqhkiG9w0BAQUFADBXMQswCQYDVQQGEwJVUzEL
MAkGA1UECAwCTVMxDzANBgNVBAcMBkJpbG94aTENMAsGA1UECgwESUVURjEbMBkG
A1UEAwwSYmlsb3hpLmV4YW1wbGUuY29tMB4XDTA1MTAyNDA2NDAyNloXDTA2MTAy
NDA2NDAyNlowVzELMAkGA1UEBhMCVVMxCzAJBgNVBAgMAk1TMQ8wDQYDVQQHDAZC
aWxveGkxDTALBgNVBAoMBElFVEYxGzAZBgNVBAMMEmJpbG94aS5leGFtcGxlLmNv
bTCBnzANBgkqhkiG9w0BAQEFAAOBjQAwgYkCgYEAv6GwWC0TD47JKwKljohjwFMS
N/5trYu4seHGqgkwLhV+XC3oEO55qrizh3yIcVSKk9n9OlANJTqNjVaHyvRuzE5F
W1TYsgJS2FAxI+11iwyAn92Ry+ha//ghe8K0Pa+JL+14iSsaHVQgnfw4Qk/RqflQ
6HvB8pqjlvyrJ3q4siMCAwEAAaOBsTCBrjAdBgNVHQ4EFgQU0Z+RL47W/APDtc5B
fSoQXuEFE/wwfwYDVR0jBHgwdoAU0Z+RL47W/APDtc5BfSoQXuEFE/yhW6RZMFcx
CzAJBgNVBAYTAlVTMQswCQYDVQQIDAJNUzEPMA0GA1UEBwwGQmlsb3hpMQ0wCwYD
VQQKDARJRVRGMRswGQYDVQQDDBJiaWxveGkuZXhhbXBsZS5jb22CAQAwDAYDVR0T
BAUwAwEB/zANBgkqhkiG9w0BAQUFAAOBgQBiyKHIt8TXfGNfpnJXi5jCizOxmY8Y
gln8tyPFaeyq95TGcvTCWzdoBLVpBD+fpRWrX/II5sE6VHbbAPjjVmKbZwzQAtpp
P2Fauj28t94ZeDHN2vqzjfnHjCO24kG3Juf2T80ilp9YHcDwxjUFrt86UnlC+yid
yaTeusW5Gu7v1g==
-----END CERTIFICATE-----
```

Bob (bob@biloxi.example.org) now wants to send a BYE request to Alice at the end of the dialog initiated in the previous example. He therefore creates the following BYE request, which he forwards to the 'biloxi.example.org' proxy server that instantiates the authentication service role:

```
BYE sip:alice@pc33.atlanta.example.com SIP/2.0
Via: SIP/2.0/TLS 192.0.2.4;branch=z9hG4bKnashds10
Max-Forwards: 70
From: Bob <sip:bob@biloxi.example.org>;tag=a6c85cf
To: Alice <sip:alice@atlanta.example.com>;tag=1928301774
Call-ID: a84b4c76e66710
CSeq: 231 BYE
Content-Length: 0
```

When the authentication service receives the BYE, it authenticates Bob by sending a 407 response. As a result, Bob adds an Authorization header to his request, and resends to the biloxi.example.org authentication service. Now that the service is sure of Bob's identity, it prepares to calculate an Identity header for the request. Note that this request does not have a Date header field. Accordingly, the biloxi.example.org will add a Date header to the request before calculating the identity signature. If the Content-Length header were not present, the authentication service would add it as well. The baseline message is thus:

```
BYE sip:alice@pc33.atlanta.example.com SIP/2.0
Via: SIP/2.0/TLS 192.0.2.4;branch=z9hG4bKnashds10
Max-Forwards: 70
From: Bob <sip:bob@biloxi.example.org>;tag=a6c85cf
To: Alice <sip:alice@atlanta.example.com>;tag=1928301774
Date: Thu, 21 Feb 2002 14:19:51 GMT
Call-ID: a84b4c76e66710
CSeq: 231 BYE
Content-Length: 0
```

Also note that this request contains no Contact header field. Accordingly, biloxi.example.org will place no value in the canonical string for the addr-spec of the Contact address. Also note that there is no message body, and accordingly, the signature string will terminate, in this case, with two vertical bars. The canonical string over which the identity signature will be generated is the following (note that the first line wraps because of RFC editorial conventions):

```
sip:bob@biloxi.example.org|sip:alice@atlanta.example.com|
a84b4c76e66710|231 BYE|Thu, 21 Feb 2002 14:19:51 GMT||
```

The resulting signature (sha1WithRsaEncryption) using the private RSA key given above for biloxi.example.org, with base64 encoding, is the following:

```
sv5CTo05KqpSmtHt3dcEiO/1CWTSZtnG3iV+1nmurLXV/HmtyNS7Ltrg9dlxkWzo
eU7d7OV8HweTTDobV3itTmgPwCFjaEmMyEI3d7SyN21yNDo2ER/Ovgtw0Lu5csIp
pPqOg1uXndzHbG7mR6Rl9BnUhHufVRbp51Mn3w0gfUs=
```

Accordingly, the biloxi.example.org authentication service will create an Identity header containing that base64 signature string. It will also add an HTTPS URL where its certificate is made available. With those two headers added, the message looks like the following:

```
BYE sip:alice@pc33.atlanta.example.com SIP/2.0
Via: SIP/2.0/TLS 192.0.2.4;branch=z9hG4bKnashds10
Max-Forwards: 70
From: Bob <sip:bob@biloxi.example.org>;tag=a6c85cf
To: Alice <sip:alice@atlanta.example.com>;tag=1928301774
Date: Thu, 21 Feb 2002 14:19:51 GMT
Call-ID: a84b4c76e66710
CSeq: 231 BYE
Identity:
  "sv5CTo05KqpSmtHt3dcEiO/1CWTSZtnG3iV+1nmurLXV/HmtyNS7Ltrg9dlxkWzo
   eU7d7OV8HweTTDobV3itTmgPwCFjaEmMyEI3d7SyN21yNDo2ER/Ovgtw0Lu5csIp
   pPqOg1uXndzHbG7mR6Rl9BnUhHufVRbp51Mn3w0gfUs="
Identity-Info: <https://biloxi.example.org/biloxi.cer>;alg=rsa-sha1
Content-Length: 0
```

biloxi.example.org then forwards the request normally.

11. Identity and the TEL URI Scheme

Since many SIP applications provide a Voice over IP (VoIP) service, telephone numbers are commonly used as identities in SIP deployments. In the majority of cases, this is not problematic for the identity mechanism described in this document. Telephone numbers commonly appear in the username portion of a SIP URI (e.g., 'sip:+17005551008@chicago.example.com;user=phone'). That username conforms to the syntax of the TEL URI scheme (RFC 3966 [13]). For this sort of SIP address-of-record, chicago.example.com is the appropriate signatory.

It is also possible for a TEL URI to appear in the SIP To or From header field outside the context of a SIP or SIPS URI (e.g., 'tel:+17005551008'). In this case, it is much less clear which signatory is appropriate for the identity. Fortunately for the identity mechanism, this form of the TEL URI is more common for the To header field and Request-URI in SIP than in the From header field, since the UAC has no option but to provide a TEL URI alone when the remote domain to which a request is sent is unknown. The local domain, however, is usually known by the UAC, and accordingly it can

form a proper From header field containing a SIP URI with a username in TEL URI form. Implementations that intend to send their requests through an authentication service SHOULD put telephone numbers in the From header field into SIP or SIPS URIs whenever possible.

If the local domain is unknown to a UAC formulating a request, it most likely will not be able to locate an authentication service for its request, and therefore the question of providing identity in these cases is somewhat moot. However, an authentication service MAY sign a request containing a TEL URI in the From header field. This is permitted in this specification strictly for forward compatibility purposes. In the longer-term, it is possible that ENUM [14] may provide a way to determine which administrative domain is responsible for a telephone number, and this may aid in the signing and verification of SIP identities that contain telephone numbers. This is a subject for future work.

12. Privacy Considerations

The identity mechanism presented in this document is compatible with the standard SIP practices for privacy described in RFC 3323 [3]. A SIP proxy server can act both as a privacy service and as an authentication service. Since a user agent can provide any From header field value that the authentication service is willing to authorize, there is no reason why private SIP URIs that contain legitimate domains (e.g., sip:anonymous@example.com) cannot be signed by an authentication service. The construction of the Identity header is the same for private URIs as it is for any other sort of URIs.

Note, however, that an authentication service must possess a certificate corresponding to the host portion of the addr-spec of the From header field of any request that it signs; accordingly, using domains like 'anonymous.invalid' will not be possible for privacy services that also act as authentication services. The assurance offered by the usage of anonymous URIs with a valid domain portion is "this is a known user in my domain that I have authenticated, but I am keeping its identity private". The use of the domain 'anonymous.invalid' entails that no corresponding authority for the domain can exist, and as a consequence, authentication service functions are meaningless.

The "header" level of privacy described in RFC 3323 requests that a privacy service alter the Contact header field value of a SIP message. Since the Contact header field is protected by the signature in an Identity header, privacy services cannot be applied after authentication services without a resulting integrity violation.

RFC 3325 [12] defines the "id" priv-value token, which is specific to the P-Asserted-Identity header. The sort of assertion provided by the P-Asserted-Identity header is very different from the Identity header presented in this document. It contains additional information about the sender of a message that may go beyond what appears in the From header field; P-Asserted-Identity holds a definitive identity for the sender that is somehow known to a closed network of intermediaries that presumably the network will use this identity for billing or security purposes. The danger of this network-specific information leaking outside of the closed network motivated the "id" priv-value token. The "id" priv-value token has no implications for the Identity header, and privacy services MUST NOT remove the Identity header when a priv-value of "id" appears in a Privacy header.

Finally, note that unlike RFC 3325, the mechanism described in this specification adds no information to SIP requests that has privacy implications.

13. Security Considerations

13.1. Handling of digest-string Elements

This document describes a mechanism that provides a signature over the Contact, Date, Call-ID, CSeq, To, and From header fields of SIP requests. While a signature over the From header field would be sufficient to secure a URI alone, the additional headers provide replay protection and reference integrity necessary to make sure that the Identity header will not be used in cut-and-paste attacks. In general, the considerations related to the security of these headers are the same as those given in RFC 3261 for including headers in tunneled 'message/sip' MIME bodies (see Section 23 in particular). The following section details the individual security properties obtained by including each of these header fields within the signature; collectively, this set of header fields provides the necessary properties to prevent impersonation.

The From header field indicates the identity of the sender of the message, and the SIP address-of-record URI in the From header field is the identity of a SIP user, for the purposes of this document. The To header field provides the identity of the SIP user that this request targets. Providing the To header field in the Identity signature serves two purposes: first, it prevents cut-and-paste attacks in which an Identity header from legitimate request for one user is cut-and-pasted into a request for a different user; second, it preserves the starting URI scheme of the request, which helps prevent downgrade attacks against the use of SIPS.

The Date and Contact headers provide reference integrity and replay protection, as described in RFC 3261, Section 23.4.2. Implementations of this specification MUST NOT deem valid a request with an outdated Date header field (the RECOMMENDED interval is that the Date header must indicate a time within 3600 seconds of the receipt of a message). Implementations MUST also record Call-IDs received in valid requests containing an Identity header, and MUST remember those Call-IDs for at least the duration of a single Date interval (i.e., commonly 3600 seconds). Because a SIP-compliant UA never generates the same Call-ID twice, verifiers can use the Call-ID to recognize cut-and-paste attacks; the Call-ID serves as a nonce. The result of this is that if an Identity header is replayed within the Date interval, verifiers will recognize that it is invalid because of a Call-ID duplication; if an Identity header is replayed after the Date interval, verifiers will recognize that it is invalid because the Date is stale. The CSeq header field contains a numbered identifier for the transaction, and the name of the method of the request; without this information, an INVITE request could be cut-and-pasted by an attacker and transformed into a BYE request without changing any fields covered by the Identity header, and moreover requests within a certain transaction could be replayed in potentially confusing or malicious ways.

The Contact header field is included to tie the Identity header to a particular user agent instance that generated the request. Were an active attacker to intercept a request containing an Identity header, and cut-and-paste the Identity header field into its own request (reusing the From, To, Contact, Date, and Call-ID fields that appear in the original message), the attacker would not be eligible to receive SIP requests from the called user agent, since those requests are routed to the URI identified in the Contact header field. However, the Contact header is only included in dialog-forming requests, so it does not provide this protection in all cases.

It might seem attractive to provide a signature over some of the information present in the Via header field value(s). For example, without a signature over the sent-by field of the topmost Via header, an attacker could remove that Via header and insert its own in a cut-and-paste attack, which would cause all responses to the request to be routed to a host of the attacker’s choosing. However, a signature over the topmost Via header does not prevent attacks of this nature, since the attacker could leave the topmost Via intact and merely insert a new Via header field directly after it, which would cause responses to be routed to the attacker’s host "on their way" to the valid host, which has exactly the same end result. Although it is possible that an intermediary-based authentication service could guarantee that no Via hops are inserted between the sending user agent and the authentication service, it could not

prevent an attacker from adding a Via hop after the authentication service, and thereby preempting responses. It is necessary for the proper operation of SIP for subsequent intermediaries to be capable of inserting such Via header fields, and thus it cannot be prevented. As such, though it is desirable, securing Via is not possible through the sort of identity mechanism described in this document; the best known practice for securing Via is the use of SIPS.

This mechanism also provides a signature over the bodies of SIP requests. The most important reason for doing so is to protect Session Description Protocol (SDP) bodies carried in SIP requests. There is little purpose in establishing the identity of the user that originated a SIP request if this assurance is not coupled with a comparable assurance over the media descriptors. Note, however, that this is not perfect end-to-end security. The authentication service itself, when instantiated at a intermediary, could conceivably change the SDP (and SIP headers, for that matter) before providing a signature. Thus, while this mechanism reduces the chance that a replayer or man-in-the-middle will modify SDP, it does not eliminate it entirely. Since it is a foundational assumption of this mechanism that the users trust their local domain to vouch for their security, they must also trust the service not to violate the integrity of their message without good reason. Note that RFC 3261, Section 16.6, states that SIP proxy servers "MUST NOT add to, modify, or remove the message body."

In the end analysis, the Identity and Identity-Info headers cannot protect themselves. Any attacker could remove these headers from a SIP request, and modify the request arbitrarily afterwards. However, this mechanism is not intended to protect requests from men-in-the-middle who interfere with SIP messages; it is intended only to provide a way that SIP users can prove definitively that they are who they claim to be. At best, by stripping identity information from a request, a man-in-the-middle could make it impossible to distinguish any illegitimate messages he would like to send from those messages sent by an authorized user. However, it requires a considerably greater amount of energy to mount such an attack than it does to mount trivial impersonations by just copying someone else's From header field. This mechanism provides a way that an authorized user can provide a definitive assurance of his identity that an unauthorized user, an impersonator, cannot.

One additional respect in which the Identity-Info header cannot protect itself is the 'alg' parameter. The 'alg' parameter is not included in the digest-string, and accordingly, a man-in-the-middle might attempt to modify the 'alg' parameter. However, it is important to note that preventing men-in-the-middle is not the primary impetus for this mechanism. Moreover, changing the 'alg'

would at worst result in some sort of bid-down attack, and at best cause a failure in the verifier. Note that only one valid 'alg' parameter is defined in this document and that thus there is currently no weaker algorithm to which the mechanism can be bid down. 'alg' has been incorporated into this mechanism for forward-compatibility reasons in case the current algorithm exhibits weaknesses, and requires swift replacement, in the future.

13.2. Display-Names and Identity

As a matter of interface design, SIP user agents might render the display-name portion of the From header field of a caller as the identity of the caller; there is a significant precedent in email user interfaces for this practice. As such, it might seem that the lack of a signature over the display-name is a significant omission.

However, there are several important senses in which a signature over the display-name does not prevent impersonation. In the first place, a particular display-name, like "Jon Peterson", is not unique in the world; many users in different administrative domains might legitimately claim that name. Furthermore, enrollment practices for SIP-based services might have a difficult time discerning the legitimate display-name for a user; it is safe to assume that impersonators will be capable of creating SIP accounts with arbitrary display-names. The same situation prevails in email today. Note that an impersonator who attempted to replay a message with an Identity header, changing only the display-name in the From header field, would be detected by the other replay protection mechanisms described in Section 13.1.

Of course, an authentication service can enforce policies about the display-name even if the display-name is not signed. The exact mechanics for creating and operationalizing such policies is outside the scope of this document. The effect of this policy would not be to prevent impersonation of a particular unique identifier like a SIP URI (since display-names are not unique identifiers), but to allow a domain to manage the claims made by its users. If such policies are enforced, users would not be free to claim any display-name of their choosing. In the absence of a signature, man-in-the-middle attackers could conceivably alter the display-names in a request with impunity. Note that the scope of this specification is impersonation attacks, however, and that a man-in-the-middle might also strip the Identity and Identity-Info headers from a message.

There are many environments in which policies regarding the display-name aren't feasible. Distributing bit-exact and internationalizable display-names to end-users as part of the enrollment or registration process would require mechanisms that are not explored in this

document. In the absence of policy enforcement regarding domain names, there are conceivably attacks that an adversary could mount against SIP systems that rely too heavily on the display-name in their user interface, but this argues for intelligent interface design, not changes to the mechanisms. Relying on a non-unique identifier for identity would ultimately result in a weak mechanism.

13.3. Securing the Connection to the Authentication Service

The assurance provided by this mechanism is strongest when a user agent forms a direct connection, preferably one secured by TLS, to an intermediary-based authentication service. The reasons for this are twofold:

> If a user does not receive a certificate from the authentication service over this TLS connection that corresponds to the expected domain (especially when the user receives a challenge via a mechanism such as Digest), then it is possible that a rogue server is attempting to pose as an authentication service for a domain that it does not control, possibly in an attempt to collect shared secrets for that domain.
>
> Without TLS, the various header field values and the body of the request will not have integrity protection when the request arrives at an authentication service. Accordingly, a prior legitimate or illegitimate intermediary could modify the message arbitrarily.

Of these two concerns, the first is most material to the intended scope of this mechanism. This mechanism is intended to prevent impersonation attacks, not man-in-the-middle attacks; integrity over the header and bodies is provided by this mechanism only to prevent replay attacks. However, it is possible that applications relying on the presence of the Identity header could leverage this integrity protection, especially body integrity, for services other than replay protection.

Accordingly, direct TLS connections SHOULD be used between the UAC and the authentication service whenever possible. The opportunistic nature of this mechanism, however, makes it very difficult to constrain UAC behavior, and moreover there will be some deployment architectures where a direct connection is simply infeasible and the UAC cannot act as an authentication service itself. Accordingly, when a direct connection and TLS are not possible, a UAC should use the SIPS mechanism, Digest 'auth-int' for body integrity, or both when it can. The ultimate decision to add an Identity header to a

request lies with the authentication service, of course; domain policy must identify those cases where the UAC's security association with the authentication service is too weak.

13.4. Domain Names and Subordination

When a verifier processes a request containing an Identity-Info header, it must compare the domain portion of the URI in the From header field of the request with the domain name that is the subject of the certificate acquired from the Identity-Info header. While it might seem that this should be a straightforward process, it is complicated by two deployment realities. In the first place, certificates have varying ways of describing their subjects, and may indeed have multiple subjects, especially in 'virtual hosting' cases where multiple domains are managed by a single application. Secondly, some SIP services may delegate SIP functions to a subordinate domain and utilize the procedures in RFC 3263 [4] that allow requests for, say, 'example.com' to be routed to 'sip.example.com'. As a result, a user with the AoR 'sip:jon@example.com' may process its requests through a host like 'sip.example.com', and it may be that latter host that acts as an authentication service.

To meet the second of these problems, a domain that deploys an authentication service on a subordinate host MUST be willing to supply that host with the private keying material associated with a certificate whose subject is a domain name that corresponds to the domain portion of the AoRs that the domain distributes to users. Note that this corresponds to the comparable case of routing inbound SIP requests to a domain. When the NAPTR and SRV procedures of RFC 3263 are used to direct requests to a domain name other than the domain in the original Request-URI (e.g., for 'sip:jon@example.com', the corresponding SRV records point to the service 'sip1.example.org'), the client expects that the certificate passed back in any TLS exchange with that host will correspond exactly with the domain of the original Request-URI, not the domain name of the host. Consequently, in order to make inbound routing to such SIP services work, a domain administrator must similarly be willing to share the domain's private key with the service. This design decision was made to compensate for the insecurity of the DNS, and it makes certain potential approaches to DNS-based 'virtual hosting' unsecurable for SIP in environments where domain administrators are unwilling to share keys with hosting services.

A verifier MUST evaluate the correspondence between the user's identity and the signing certificate by following the procedures defined in RFC 2818 [11], Section 3.1. While RFC 2818 deals with the use of HTTP in TLS, the procedures described are applicable to

verifying identity if one substitutes the "hostname of the server" in HTTP for the domain portion of the user's identity in the From header field of a SIP request with an Identity header.

Because the domain certificates that can be used by authentication services need to assert only the hostname of the authentication service, existing certificate authorities can provide adequate certificates for this mechanism. However, not all proxy servers and user agents will be able to support the root certificates of all certificate authorities, and moreover there are some significant differences in the policies by which certificate authorities issue their certificates. This document makes no recommendations for the usage of particular certificate authorities, nor does it describe any particular policies that certificate authorities should follow, but it is anticipated that operational experience will create de facto standards for authentication services. Some federations of service providers, for example, might only trust certificates that have been provided by a certificate authority operated by the federation. It is strongly RECOMMENDED that self-signed domain certificates should not be trusted by verifiers, unless some previous key exchange has justified such trust.

For further information on certificate security and practices, see RFC 3280 [9]. The Security Considerations of RFC 3280 are applicable to this document.

13.5. Authorization and Transitional Strategies

Ultimately, the worth of an assurance provided by an Identity header is limited by the security practices of the domain that issues the assurance. Relying on an Identity header generated by a remote administrative domain assumes that the issuing domain used its administrative practices to authenticate its users. However, it is possible that some domains will implement policies that effectively make users unaccountable (e.g., ones that accept unauthenticated registrations from arbitrary users). The value of an Identity header from such domains is questionable. While there is no magic way for a verifier to distinguish "good" from "bad" domains by inspecting a SIP request, it is expected that further work in authorization practices could be built on top of this identity solution; without such an identity solution, many promising approaches to authorization policy are impossible. That much said, it is RECOMMENDED that authentication services based on proxy servers employ strong authentication practices such as token-based identifiers.

One cannot expect the Identity and Identity-Info headers to be supported by every SIP entity overnight. This leaves the verifier in a compromising position; when it receives a request from a given SIP

user, how can it know whether or not the sender's domain supports Identity? In the absence of ubiquitous support for identity, some transitional strategies are necessary.

A verifier could remember when it receives a request from a domain that uses Identity, and in the future, view messages received from that domain without Identity headers with skepticism.

A verifier could query the domain through some sort of callback system to determine whether or not it is running an authentication service. There are a number of potential ways in which this could be implemented; use of the SIP OPTIONS method is one possibility. This is left as a subject for future work.

In the long term, some sort of identity mechanism, either the one documented in this specification or a successor, must become mandatory-to-use for the SIP protocol; that is the only way to guarantee that this protection can always be expected by verifiers.

Finally, it is worth noting that the presence or absence of the Identity headers cannot be the sole factor in making an authorization decision. Permissions might be granted to a message on the basis of the specific verified Identity or really on any other aspect of a SIP request. Authorization policies are outside the scope of this specification, but this specification advises any future authorization work not to assume that messages with valid Identity headers are always good.

14. IANA Considerations

This document requests changes to the header and response-code sub-registries of the SIP parameters IANA registry, and requests the creation of two new registries for parameters for the Identity-Info header.

14.1. Header Field Names

This document specifies two new SIP headers: Identity and Identity-Info. Their syntax is given in Section 9. These headers are defined by the following information, which has been added to the header sub-registry under http://www.iana.org/assignments/sip-parameters.

Header Name: Identity
Compact Form: y
Header Name: Identity-Info
Compact Form: n

14.2. 428 'Use Identity Header' Response Code

This document registers a new SIP response code, which is described in Section 6. It is sent when a verifier receives a SIP request that lacks an Identity header in order to indicate that the request should be re-sent with an Identity header. This response code is defined by the following information, which has been added to the method and response-code sub-registry under http://www.iana.org/assignments/sip-parameters.

Response Code Number: 428
Default Reason Phrase: Use Identity Header

14.3. 436 'Bad Identity-Info' Response Code

This document registers a new SIP response code, which is described in Section 6. It is used when the Identity-Info header contains a URI that cannot be dereferenced by the verifier (either the URI scheme is unsupported by the verifier, or the resource designated by the URI is otherwise unavailable). This response code is defined by the following information, which has been added to the method and response-code sub-registry under http://www.iana.org/assignments/sip-parameters.

Response Code Number: 436
Default Reason Phrase: Bad Identity-Info

14.4. 437 'Unsupported Certificate' Response Code

This document registers a new SIP response code, which is described in Section 6. It is used when the verifier cannot validate the certificate referenced by the URI of the Identity-Info header, because, for example, the certificate is self-signed, or signed by a root certificate authority for whom the verifier does not possess a root certificate. This response code is defined by the following information, which has been added to the method and response-code sub-registry under http://www.iana.org/assignments/sip-parameters.

Response Code Number: 437
Default Reason Phrase: Unsupported Certificate

14.5. 438 'Invalid Identity Header' Response Code

This document registers a new SIP response code, which is described in Section 6. It is used when the verifier receives a message with an Identity signature that does not correspond to the digest-string calculated by the verifier. This response code is defined by the following information, which has been added to the method and response-code sub-registry under http://www.iana.org/assignments/sip-parameters.

Response Code Number: 438
Default Reason Phrase: Invalid Identity Header

14.6. Identity-Info Parameters

The IANA has created a new registry for Identity-Info headers. This registry is to be prepopulated with a single entry for a parameter called 'alg', which describes the algorithm used to create the signature that appears in the Identity header. Registry entries must contain the name of the parameter and the specification in which the parameter is defined. New parameters for the Identity-Info header may be defined only in Standards Track RFCs.

14.7. Identity-Info Algorithm Parameter Values

The IANA has created a new registry for Identity-Info 'alg' parameter values. This registry is to be prepopulated with a single entry for a value called 'rsa-sha1', which describes the algorithm used to create the signature that appears in the Identity header. Registry entries must contain the name of the 'alg' parameter value and the specification in which the value is described. New values for the 'alg' parameter may be defined only in Standards Track RFCs.

Appendix A. Acknowledgements

The authors would like to thank Eric Rescorla, Rohan Mahy, Robert Sparks, Jonathan Rosenberg, Mark Watson, Henry Sinnreich, Alan Johnston, Patrik Faltstrom, Paul Kyzviat, Adam Roach, John Elwell, Aki Niemi, and Jim Schaad for their comments. Jonathan Rosenberg provided detailed fixes to innumerable sections of the document. The bit-archive presented in Appendix B follows the pioneering example of RFC 4475 [16]. Thanks to Hans Persson and Tao Wan for thorough nit reviews.

Appendix B. Bit-Exact Archive of Examples of Messages

The following text block is an encoded, gzip-compressed TAR archive of files that represent the transformations performed on the examples of messages discussed in Section 10. It includes for each example:

- (foo).message: the original message
- (foo).canonical: the canonical string constructed from that message
- (foo).sha1: the SHA1 hash of the canonical string (hexadecimal)
- (foo).signed: the RSA-signed SHA1 hash of the canonical string (binary)
- (foo).signed.enc: the base64 encoding of the RSA-signed SHA1 hash of the canonical string as it would appear in the request
- (foo).identity: the original message with the Identity and Identity-Info headers added

Also included in the archive are two public key/certificate pairs, for atlanta.example.com and biloxi.example.org, respectively, including:

- (foo).cer: the certificate of the domain
- (foo).privkey: the private key of the domain
- (foo).pubkey: the public key of the domain, extracted from the cert file for convenience

To recover the compressed archive file intact, the text of this document may be passed as input to the following Perl script (the output should be redirected to a file or piped to "tar -xzvf -").

```
#!/usr/bin/perl
use strict;
my $bdata = "";
use MIME::Base64;
while(<>) {
 if (/-- BEGIN MESSAGE ARCHIVE --/ .. /-- END MESSAGE ARCHIVE --/) {
     if ( m/^\s*[^\s]+\s*$/) {
         $bdata = $bdata . $_;
     }
  }
}
print decode_base64($bdata);
```

Alternatively, the base-64 encoded block can be edited by hand to remove document structure lines and fed as input to any base-64 decoding utility.

B.1. Encoded Reference Files

```
-- BEGIN MESSAGE ARCHIVE --
H4sICFfaz0QCA25ld2lkZW50LnRhcgDsW0us5NhZ7gUSwqiF2CAhFikiIQhFt992
+U46it+u8qPK5Uc9WPlVfj/KdpXtomEDCxaAhFggISE2WSHCIoIFioQQC8gqAhRA
QQTY8JJAbMgGIYTv7b7T09PT0xNl+mqS3F8qVd3jY/uc85//+87/nXOLoIv9oGjB
B2/PIAiDSBwfv1GERInxG8EwAh6/37UHMIQRKIljCI4+gGCUGKtP8Ad3YKemderJ
5EFSBW1QN2Xxmnp5GtblqXqUPfIffBdZcet/p82conUee0H9sfsfhiACw17nfwQa
y+Dra+MkQGFkrI+TOPJgAt37/63bo2tjeHGuTVh+bc6FOUub/E0poM7nLGqyLJ06
Id3NGTocPxytMWF6jNJYpDqIoXVLoDlmr+pNx+o7ztZ1ke8WtnXhFUClU5GGLZ6l
O3YN8T3P0Usm1GyG9lQGEiBXFE6+yPecSSvPykuV4TPB5ne9xNEO8KxQVXnk3cqn
/TaK3C3T7A08cRGokyJPUzmrV7k5pHK7i5bQyOambNcDLxUmH9zMD2sl8FGa+WGt
BG6bGe5nHafvFnK5n0dnT6N1nmF0mgt3EK3OxQVdiuMzZrNOhPxNOF37W7w4LmsL
OA0Mpeqt7RTKTrDX1CztZgezbM7rLlvQeBnhWzWOV5qDZEdMahLZTo8Wq0oZOL4X
FgkgMhY4pNBdU53sHVvlaIX5TjqH0+JkYXAXmmzgSI7H9N3RvHingrIOAUIzCph4
GhsdHGDwET+WCO5SuDtwxXKNvneGYrWiQ5WhaTEJXb0LXb6Trgd2DS0ZZscLWm6B
au3aO48HZK4GEWgzN2oRTuBaG/vLXA+aZKh8kDBYyJj7bHWREXgjMWxIgFQrxPyx
b3eUc3EEH6iEptuYL1zFRCpr22rPXujFs9EPx0s+o67pbhzRa/eOjvEZX+wjt1hH
gKpDHdvdXJA5er1Y22tRXXed+KwyxzFadFtZyW1st4E7V7ROO4Rqw5Cnx6ncXb/Z
5ztdUOmx34dX3Ck8cydPc76+a5uO4XLTMI9Q3iIwDJBOloNbUahd5OK7FnQu637t
L/cQdlSHel5tRVjh84Jfhl7pDfV2zZyPeEVs3D3t8XoKAVzDo3YAad6sp4r8nCUb
UmxUUWAL9lRiS848gHAm+nZNcQF78RIY2lk6qq6DnFO30Q4B2JaLG2WTkcZ2uVx7
ezqGS4vqngA30c5r3KsI8ODevsvtFf6v6vicBsMd8j+ME+Qt/0PjAnCsT5AQes//
d8z/a4OerNZze4z+iczvXqwBtvrI+7TMhDq3WqlMK9nlKt3a0z2RHGGlCQ8jMtub
akAY2zocFupKgghFgbyFoS8BZx7Yl3mZXDZt5ZwYcj5kezmjEwY/YCO4rk+lFQc+
26mK7GYb+rhviUDaVKy2X5DZUvOAOd8VeYQUtOfJ6QxVKtCW0DakDRBDOb3cIk3h
F7toGs5wBFldupDkxU1TXS7dnKN1mgFumFWGNmhb8AJH0omt08VC23Jtj1O0A9sn
ZMFvA6KMp8s6FYZmkbj7RdcoudzWYdsCq+3SmrVIvq9iqJOxaIu1+6ho406UU2vF
ohHFJNVUDOr4sEIxeK0O6nJKHFZhclxeLK4DpvUqSdSqG1+eerx35ELXrPfF5gzq
BWs4joD2qSUehFTp8aXsremUp0mrLxp+tnVMFALaFWhZHg6HWorIohz2um5KZcV4
QUcNh4BdC9HZV8ikckSn5WM83neiONKavbQlS4MlANoplaQn67JbMLQ2XSPumQa1
```

```
OD9iBLYPiyDjudXR4en9xuHQdHmIDGp6VsjyyBvTE85DwIJMty65T2PDtkJqa4Gz
Va/KPcjRF8i38qUytVhdmrEUb1rqHDnx7lFyGd+2RC1FCYwFOMErfKO3oymKyceF
n8Q7oyfs1eqMEFsqJw1oOfhmaoQNCmJluerLmeSox20+g1idmdZA7zKolVXLMvKY
TpCp3KwzlSHYhjpmBCGHXZEp1CnlI0nalZdxHPxtUDLsEFlNGfqGBRCgY9CCd97w
YpuQ4HlY8Kyus6wBZ3LIb0tNXx2XmpOdd9EwqPv1VlB8Dgvdbr2S4dNWBnZVirLp
Qbgsh0MSKJ646reXI3K8nKSLaHL9nlrRQdVtsbWRviDVDwyrTzD+n9yPGf7fhP8j
5kO3+I/AN/k/gZHYPf7fMf6vLEaZs++FfvGg0pDIGkfRmLsj2PLX6R5NY6JGcywT
6x9OCcDrOOGjUgLwOk74qJQAvJYT3o3O93f6e3b958ZZ2cdvQ/55s/6DvEf/QbBr
/YeAifv4/yToP3DCsnQyfZP+s32j/mOO6Tp3ub75uf6TLipXpDDH5DWVbp7VCzve
sGxrnfDuWEEErgvprjN2eda4aFS9PzVXGWzLmTSsmvSgcTQyfgYtK6/LkOsy4D2F
nX15k4AAm6p+k9Y/FxD2LOBs+nMgph+o/YgXev+u9pM/746BZ4EotJ7YZ0qunQHX
ZJni8v5B4wWaXjKJTnfhLmWvRYMzIXYbFjI5jFzInZwlZZR0gmoAGoi39e6ENYEk
HsO0UyJ7umXRkl/i+LGOLxE6zD3bkFOqoJYZrS3Mo5bYjjSc16cLjwvABjZ3Tbgw
EIHu51MYjruBLihkPUwjBwTDKJjJ0MqZLpQpjMVG40i2HhaHDtNTcH08ZDpASGdm
Vh2T7DzUC/SINbE6epSnaWfJNGP36oT2b+QcHeOFULeg/XStYOQGpFdc6+EMcDBK
fXviBR7sukN3IxIljBR2fkm/UvlF3SHaEOu9Kng98MJNO5PObPM9s20E9IU2zrbV
NVXduLbrRP35fLmVfYCXdZ9mrHGr+yzi5y5+n7CIsCNRdBx901oTYGirG/vMgJcP
mP/XeqHOxIMszduZuT2I2qEqFtsYT9j4suzz3WwHhFkxa4eV4ATDkcJN0Tub7Obi
l4xiVww3PVTrTb0F53O84Qlbcl16TBnsXHb33UWn26oCVojgnBJk1lLYPuAkDTkf
L8mhkBJ2iWCpiC5OB8ScQXFWUTvJ47o+sYS6nRFWkbHTIfaBwTGDU7PBxRN5hsMn
97rPvb3K/29B/nmz/kOit/wPI+NaYFz/49j9/s8nR/8Jb/UfFixdZqes1VXSpDV9
3CxjcUVb/RwFc6SNybjHPOfImvRJ2OKeEoQ6QBb58aQspcM86u350UQOEGHRULYs
Ec0uDzIlkqqZ2q6txQOdKTuL4xNyu1G4OXtA95ICEEINTlmB7GqdqrH0TG7jhdyX
vs2yPshFrEmJ1dTmymAmDflxuQHlpgjqeJi/pP8syEMjzOWtnCabMJmljbhsIwM1
CpjqVwY78D7TH/gcWSUkqF0uQRaDK2/pxB6UAouR+r3iqCEHiQ/mogxSvcX05ukQ
6jt7cTwPEr9uiHq7BWMT2xU51cIUhPOxTu0rqannADguEKwdDeu1GNJz6bxXbOVy
nFKywvH7qaS7J1ZZbIUp4WYQ7+LMtf5DoESp0loF6Q4K5LsNryOnNhebXZ9ujcPA
uPDMZJcd2w5Q4TNrBLsMy4WAaO7eoGbKZSo6CB4d5mIHLiQZKDjKXfKzmXWj/zBr
o/IxNzemOTZbgzDarnmDbqXj4GtxsYVSA1xHnVSTeSqZFpqCKiD0etuj2BwV5Yuz
79UCoglCNqgzaEh+IUyD1Y2YIgak3kTDfnaKW2XV7jkvYzcRL0vAkdal3OL3Z0tA
bEmp3VOqKMtQsmpJcxDMmytnzEcHh7WtoB1yzTsNZhfJCYJ1Ap3SS+ACJj3MV5mG
Rp0y1Zos25ebOT47nU8kSB8RD/UuR8cWGddFYbKR2F0op5BLi2jaLdE8BigUVLYb
E/b8eGdXOeNJ3M1I51WYCsm035/wcEMbO/yUnKcCq66gTedIeGQW29O0lQNgtUB9
ZL7Yy71YZETcymuNFIN1RK0MGUr3Y5osBHZ9bhaYVlYvEewnVwN6Bf8/fvnnW9N/
yBv9B8Wge/z/jtB/Xk8JwOs44aNSAvA6TviolAC8lhPu9Z9X4n8IHntOURax52R3
G//jAvD5+S8MxbGb9R8K38f/nVgTV1du6X7+OfwHvZNXWfC4rMOn15ecLPaCz9/u
Ddxe9cr8qTPDXMwjiYAgRtx+iqDwhNnxT83o9DMTBJ4IgTtBRkdPYOwKpq5weCKq
5tOn9wnXJzn+b37F7cdM/2/M/2AUe3H+E7vZ/0eg+/2fO7ExZicvAr3yUPTxB0T7
xJivQOQx9BCwY+fq9i/QVIwJTI2/HiOPsXfc2im86MmFikTMlQunifwGHm9Rnf6R
UNadU/vN1YQcS4S6zK8mTOlOPvt6/PncO60TPnEIb4Z7h4eAWV5N6OtGPrvntcD0
7LaxVTMUgkkSewhwThtcTT4UmB4CrJNlj+bc1eRlXBsvGMHxavIc3h4C8+chcjX5
dHPGWbOEcPlYGXkrtajv8fEShNmNaezbQkRjewoX+alWtjYo5e2gGaTS1iHlZ326
uZQPgckLCyzSJ5f2TOoC0+RK10bj1szDVccKicPn6sDPUZ80Bg2BB40rEX4NLs9h
20HKCfeaefXSw6rVcRnCp23hXyRXJPM1sc4oprAi6XSw126Fw2qBdlB4sJonn37R
p0fz4jCO8mejtq2aKxB81Sfv2SX63DtOFj6pG+dREznwOE5l0Y6PeaQERdhGV5Nx
6O7R9TsM//OgaZwwuOP9Pwh7cf57hH7i5vw3gd/j/z3+fyz4/1Gh/XsSwV6K/2sk
fwvveFP8QyRxm/9hY43r+Efg+/Ofd2KGRMM/9VLu/5knkwM5IyjUP6A4jPuI5wfU
GEw4jsEocX2ghnQdGMbgA3bP8N9l8R+HReDfefwj7/7/H0ZCOPHs/A95H/93YV/6
```

```
P0b7Veqnf3f9W3/5n9/42+/75f/65g/4f3X4+p/9w0/8wt8Mv/97f/jX/zt88Stf
+/Ljv/unb379+OvZvw3aN/7jn59+6vt/Q7n6sU3/RS36oT/5cS+a/8pXGLL7gy+R
eY1dET/8qa/+8Q9Wf/HlP6r/9DNf+J9f+8Wf/c3f/vs/z4p/Eb8Q/PePfu2Xfu53
rB/59381fvIfH05+Xr6PwE9c/D8OCu9u4/+F/nt9BOBG/yXuz//djf77bYoYwLcr
XADfilhxv+B4a/EfF+e4fTtbQG+Kfxy6Pv+D4SiMosTN+V9yzAnu4/9O4v9DN3k+
ZHfoffs/6JgQ4NRkrtlz84N2gdArCLmC0JtdoDfrDU/PT8bsu3xiNUFN/3875/Pa
NBiH8Yt6CBS0Q2SDYcYEkSl9k75Nmkmn7ebWde2WLm3646Jp2q7FtU2btq496EGc
KMgu4sH5a4dN8NccCMLYP6AMwcv+Bg/e1NMuZimTdlvXyWxx4/s5pQ0N5SXPk/d9
nrclaSuHrBhbaKb6cHiUHOYxWe8SBkK1CTFVTWbSpDDAGwjZ1vATeRvaWPWnbFIh
msyQmKNYmhz38Sa7yG+ckGy5vJKSlF5E8v0ev8mq3bwHPCTYqv9mVEAN9//p+Z+m
f9qCMMvqv/+k4fnfEiqCJbcJfVPnuyR/9XS0YxBorSR4jTK/zWywKUlfjUftlEvW
a4qqzKsSE0pyvrf629Ubir6awigcGnVEnP0IiZ5wjr4ezjNiqr/IZ9IBl2eo6PU5
BrITiUwg5p5yxcsOWqKUKXvOLE7kHEhQBbtU0/Ek4+p4NDnGZ7zh0FiJvpETJxKF
hKx6Is6AXxicGmYUJmvxjXmDTk+qzBSuZMxq0aUKTszlE6WhdM3FBkU5XZLCPT2l
8UlHKOT1ubOBsqtnREzwI5G436TkSgkxzYVkxr9bYbTDCFT/r0y9yshXUrRhlxRF
G0sprxm2SY0q2/NYCrMGwkDAo6GZ/t+MCqhh/4/MVf2Pvv7DDMz/wP8Pg/+DyQEH
yP+bUQE23P+JqD/zfxpZ9P5fewv8vwXo/d/W7OecjaRZhGWaZq04LtGUjCPIwkUQ
krUXmI1xEstIUQmbOVD/IdN/EyrAPfZ/Ff2z+v5P7RD03wpit+2TyoevQvtisv3j
fJz48e1pxN3xs+1I74vpO89MxqurnY/XnlxeLFx702lcIjvurZ8ods/MHQtevPD+
bbBr+dR5amnN25XtflV+/fCLPbs62/fO+OD7yqzx9EzqbtfLk4GznxZurp+JHZ0+
7l5+tPr8vtj2OfXr0sLKnHgrqM6DAv9H/f/bCnCP/Z+ufzOm9PyfhfVfS9hvJkXs
N4ci/iZ7gtkGAAAAAAAAAAAAAAAAAAAAAAAAAABAPX4DY+BfEQB4AAA=
-- END MESSAGE ARCHIVE --
```

Appendix C. Original Requirements

The following requirements were crafted throughout the development of the mechanism described in this document. They are preserved here for historical reasons.

- The mechanism must allow a UAC or a proxy server to provide a strong cryptographic identity assurance in a request that can be verified by a proxy server or UAS.
- User agents that receive identity assurances must be able to validate these assurances without performing any network lookup.
- User agents that hold certificates on behalf of their user must be capable of adding this identity assurance to requests.
- Proxy servers that hold certificates on behalf of their domain must be capable of adding this identity assurance to requests; a UAC is not required to support this mechanism in order for an identity assurance to be added to a request in this fashion.
- The mechanism must prevent replay of the identity assurance by an attacker.
- In order to provide full replay protection, the mechanism must be capable of protecting the integrity of SIP message bodies (to ensure that media offers and answers are linked to the signaling identity).
- It must be possible for a user to have multiple AoRs (i.e., accounts or aliases) that it is authorized to use within a domain, and for the UAC to assert one identity while authenticating itself as another, related, identity, as permitted by the local policy of the domain.

References

Normative References

[1] Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., Handley, M., and E. Schooler, "SIP: Session Initiation Protocol", RFC 3261, June 2002.

[2] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[3] Peterson, J., "A Privacy Mechanism for the Session Initiation Protocol (SIP)", RFC 3323, November 2002.

[4] Rosenberg, J. and H. Schulzrinne, "Session Initiation Protocol (SIP): Locating SIP Servers", RFC 3263, June 2002.

[5] Peterson, J., "Session Initiation Protocol (SIP) Authenticated Identity Body (AIB) Format", RFC 3893, September 2004.

[6] Crocker, D. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", RFC 4234, October 2005.

[7] Housley, R., "Cryptographic Message Syntax (CMS) Algorithms", RFC 3370, August 2002.

[8] Josefsson, S., "The Base16, Base32, and Base64 Data Encodings", RFC 3548, July 2003.

[9] Housley, R., Polk, W., Ford, W., and D. Solo, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 3280, April 2002.

[10] Housley, R. and P. Hoffman, "Internet X.509 Public Key Infrastructure Operational Protocols: FTP and HTTP", RFC 2585, May 1999.

[11] Rescorla, E., "HTTP Over TLS", RFC 2818, May 2000.

Informative References

[12] Jennings, C., Peterson, J., and M. Watson, "Private Extensions to the Session Initiation Protocol (SIP) for Asserted Identity within Trusted Networks", RFC 3325, November 2002.

[13] Schulzrinne, H., "The tel URI for Telephone Numbers", RFC 3966, December 2004.

[14] Faltstrom, P. and M. Mealling, "The E.164 to Uniform Resource Identifiers (URI) Dynamic Delegation Discovery System (DDDS) Application (ENUM)", RFC 3761, April 2004.

[15] Peterson, J., "Retargeting and Security in SIP: A Framework and Requirements", Work in Progress, February 2005.

[16] Sparks, R., Ed., Hawrylyshen, A., Johnston, A., Rosenberg, J., and H. Schulzrinne, "Session Initiation Protocol (SIP) Torture Test Messages, RFC 4475, May 2006.

Authors' Addresses

Jon Peterson
NeuStar, Inc.
1800 Sutter St
Suite 570
Concord, CA 94520
US

Phone: +1 925/363-8720
EMail: jon.peterson@neustar.biz
URI: http://www.neustar.biz/

Cullen Jennings
Cisco Systems
170 West Tasman Drive
MS: SJC-21/2
San Jose, CA 95134
USA

Phone: +1 408 902-3341
EMail: fluffy@cisco.com

Full Copyright Statement

Intellectual Property

Acknowledgement

Funding for the RFC Editor function is provided by the IETF Administrative Support Activity (IASA).