SIP WG
Internet-Draft
Expires: September 2, 2005

J. Peterson
NeuStar
C. Jennings
Cisco Systems
March 2005

Enhancements for Authenticated Identity Management in the Session Initiation Protocol (SIP)
draft-ietf-sip-identity-05

Status of this Memo

Copyright Notice

Abstract

The existing security mechanisms in the Session Initiation Protocol 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 [1]). An identity, for the purposes of this document, is defined as a SIP URI, commonly a canonical AoR employed to reach a user (such as 'sip:alice@atlanta.example.com').

RFC3261 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.

RFC3261 specifies a number of security mechanisms that can be employed by SIP UAs, including Digest, 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 pre-arranged 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 their displayed Caller-ID is accurate.

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 RFC2119 [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 which 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 RFC3261 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 address-of-record SIP 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 their 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 address-of-record 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 are can prove you are eligible to register in a domain under a particular address-of-record, you can prove you can legitimately receive requests for that address-of-record, and accordingly, when you place that address-of-record 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 their use of the From header field has been authorized. This draft addresses how that imprimatur of authentication can be shared.

RFC3261 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 [10]. 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 which 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 hold their own certificates; 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 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 certificates 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. Requirements

This draft addresses the following requirements:

- 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) which 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.

5. 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 which 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 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) which 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 validates that the domain indicated by the host portion of the AoR in the From header field authenticated the user, and permitted them to assert that From header field value. This same validation operation may be performed by Bob's UAS.

6. 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, and 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 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. Similarly, authentication services MUST add a Content-Length header field to SIP requests if one is not already present; this can help the verifier 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 performs 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 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. 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 12 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 by 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 which 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 which 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 which might legitimately be asserted for that 'username'. For example, the username can correspond to the 'private identity' as defined in 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 with the reason phrase "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 14.2.

Step 3: The authentication service SHOULD ensure that any pre-existing 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 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.

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 Section 10.

Finally, the authentication service MUST forward the message normally.

6.1 Identity within a Dialog and Retargeting

Retargeting is defined as the alteration of the Request-URI by intermediaries in order to point to another URI that corresponds to a user that cannot authenticate itself with the identity originally present in the Request-URI. By this definition, retargeting excludes translation of the Request-URI to a registered contact of an endpoint that has authenticated itself as that user.

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 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 [14].

7. Verifier Behavior

This document introduces a new logical role for SIP entities called a

'verifier', which may be instantiated by a user agent or proxy 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 which 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 recipient, 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 client processes this certificate in the usual ways, including checking that it has not expired, that the chain is valid back to a trusted CA, and that it does not appear on revocation lists. Once the certificate is acquired, it MUST be validated. 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 compare the identity of the signer with the domain portion of the URI in the From header field. The verifier MUST follow the process described in Section 14.4 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 10. If a verifier determines that the signature on the message does not correspond to the reconstructed digest-string, then a 428 'Invalid Identity Header' response MUST be

returned.

Step 4: The verifier MUST validate the Date, Contact and Call-ID headers the manner described in Section 14.1; recipients that wish to verify Identity signatures MUST support all of the operations described there.

8. 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 14.3 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 which 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 12).

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.

9. Considerations for Proxy Server

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.

10. 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:

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

Identity-Info = "Identity-Info" HCOLON ident-info (* SEMI identi-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 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 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 3261. RFC 3261 specifies that the BNF for weekday and month are 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 letter 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 14 and [5]. The precise formulation of this digest-string is, therefore (following the ABNF [6] in RFC3261):

```
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 15.6 for more information. It is MANDATORY for all implementations of this specification to 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 in placed int the Identity header field. For detailed examples of the usage of this algorithm, see Section 11.

Note on the use of 'rsa-sha1': The raw signature will result in about 170 octets of base64 encoded data (without base64, as an aside, it would be about 130 bytes). For comparison's sake, a typical HTTP Digest Authorization header (such as those used in RFC3261) with no cnonce is around 180 octets. From a speed point of view, a 2.8GHz Intel processor does somewhere in the range of 250 RSA 1024 bits signs per second or 1200 RSA 512 bits signs; verifies are roughly 10 times faster. Hardware accelerator cards are available that speed this up.

The 'absoluteURI' portion of the Identity-Info header MUST contain either an HTTP or HTTPS URI which dereferences to a resource that contains a single MIME body containing the certificate of the authentication service. These URIs MUST follow the conventions of RFC2585 [11] and 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 [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.

11. 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 generated 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 14.4 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.

11.1 Identity-Info with a Singlepart MIME body

Consider the following private key and certificate pair assigned to 'atlanta.example.com'.

```
-----BEGIN RSA PRIVATE KEY-----
MIICXQIBAAKBgQC8HmM8b9E4WNhb7tZAoBVSkKyV9rAEX3nyQbg4hXte1oW1BxC+
43MQHrG3nk6Kc9afPR6VloKwWoUoAcCnbTJ/zEiZ6dq+C5EsQGIOowYkSgqdO2po
joCnRgzgjgvAl41R2J6CE1kMwOQxNCxPnTco8l8UGdKbNLXIuNdUM1MG8QIDAQAB
AoGAAtPOGAVyNo+XSOJxE+2UBHaqMWLQyHAK7Coys57F+OnufocJqGTQwOhFMYZO
leQh0KjhgcwOUMo7gBtuotWQUbbLHTGKXiBR6Pqbm6CvhwJSuNYv0vONuTb1SMll
Kadg43na4B9kQeytn1y6lfkTkK2oYqkDVZ2AAmLSLrfhl1UCQQDp7VFItgmnybwK
PKwJs8gnF+u+K9j+sac/3vgGgrOvpxVqwoMXl6eWN//pZ/cqshanDLmtr9ahjWCD
DxYVyklrAkEAzd6JLJAhG8cZymVCS5Jf0F7FAVxpx0BgRPHwJliyUg6O4jPY+ASg
cLP6nz9a38wWZQj6rRygffGZHXbBFm+8EwJBAJmZEf5ESSK6+5VdMTlNqubAdjJw
aBMUY1U0+naL66AyfYWUIq+jDI8+RfLkKQ8H0IfvexvokW2SfwSPK1kzcfECQD/O
MQW2xgwt8ThhmeKCQ1/5f2WklsRCl5PGyH+aDeqQyIgjOaPlCzTjE1I3+JpUTryR
w9/Td4qRTrtrCv1BNDECQQCgHIzF8LFtI003w9MAEAoCyDbtHFPEj71b+qG22Yc4
SPFBAbo3JGO+mrB0MX/GwJr+3DfgzMHaUx/tinPr+u1D
-----END RSA PRIVATE KEY-----
-----BEGIN CERTIFICATE-----
MIIC/TCCAmagAwIBAgIBADANBgkqhkiG9w0BAQQFADBZMQswCQYDVQQGEwJVUzEQ
MA4GA1UECBMHR2VvcmdpYTESMBAGA1UEBxQJQXRsYXQIbnRhMQ0wCwYDVQQKEwRJ
RVRGMRUwEwYDVQQLFAxTT0lQCAgISVAgV0cwHhcNMDQwOTEzMTAxMzAzWhcNMDUw
OTEzMTAxMzAzWjBZMQswCQYDVQQGEwJVUzEQMA4GA1UECBMHR2VvcmdpYTESMBAG
A1UEBxQJQXRsYXQIbnRhMQ0wCwYDVQQKEwRJRVRGMRUwEwYDVQQLFAxTT0lQCAgI
SVAgV0cwgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBALweYzxv0ThY2Fvu1kCg
FVKQrJX2sARfefJBuDiFe17WhbUHEL7jcxAesbeeTopz1p89HpWWgrBahSgBwKdt
Mn/MSJnp2r4LkSxAYg6jBiRKCp07amiOgKdGDOCOC8CXjVHYnoITWQzA5DE0LE+d
NyjyXxQZ0ps0tci411QzUwbxAgMBAAGjgdQwgdEwHQYDVR0OBBYEFGfCU7cNxqSK
NurvFqz8gj5px8uoMIGBBgNVHSMEejB4gBRnwlO3Dcakijbq7xas/II+acfLqKFd
pFswWTELMAkGA1UEBhMCVVMxEDAOBgNVBAgTB0dlb3JnaWExEjAQBgNVBAcUCUF0
bGF0CG50YTENMAsGA1UEChMESUVURjEVMBMGA1UECxQMU09JUAgICElQIFdHggEA
MAwGA1UdEwQFMAMBAf8wHgYDVR0RBBcwFYITYXRsYW50YS5leGFtcGxlLmNvbTAN
BgkqhkiG9w0BAQQFAAOBgQAc0a/5hU6yqRTxwqoBuRk/iSqDnJD/B0QQnSFLqdjy
QV/Pm+aluA05aLRDWq6w/ufwX2HPLOvXYubpnNzjpaWCx3OLr4b5NwnsfNSxtKBJ
vI9PWwhSW6VMo/cT2llhNudCmN+LXPd/SLy3gnGvXtwcrWAT8MVYmkCUQTRvbWaR
fQ==
-----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.exmple.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, in 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|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:

```
NJguAbpmYXjnlxFmlOkumMI+MZXjB2iV/NW5xsFQqzD/p4yiovrJBqhd3TZkegns
moHryzk9gTBH7Gj/erixEFIf82o3Anmb+CIbrgdl03gGaD6ICvkpVqoMXZZjdvSp
ycyHOhh1cmUx3b9Vr3pZuEh+cB01pbMQ8B1ch++iMjw=
```

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:

```
        INVITE sip:bob@biloxi.exmple.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:"NJguAbpmYXjnlxFmlOkumMI+MZXjB2iV/NW5xsFQqzD/
p4yiovrJBqhd3TZkegn
                 smoHryzk9gTBH7Gj/
erixEFIf82o3Anmb+CIbrgdl03gGaD6ICvkpVqoMXZZjdvS
                 pycyHOhh1cmUx3b9Vr3pZuEh+cB01pbMQ8B1ch++iMjw="
        Identity-Info: <https://atlanta.example.com/cert>;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 de-references the URL 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 de-referencing the Identity-Info header, Bob can verify that the given set of headers and the message body have not been modified.

11.2 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-----
MIICXQIBAAKBgQDDIREMIIS9vBBET2FFHss2Lbwri/nK+AMoUZ74UT3amG/bYgDn
H86eUUEjGfV3cfXErFXSnI86sUALoKjjwGYBoiUuaMhyerZyF+D9St2plnBeq6fq
rbaPpL6bvIAF636/O2+GFP3LSLj6KS4HQwnsaUBr2YzykBD05PfwrH28VQIDAQAB
AoGAZLRJFwglWcKYZpjNK54T5HdAGP1Zwo2zG3jcYW2UTZ/EguWWb7HzsbNfuZzp
GWcgHwuOE28nYHQgCKA26avfOGuebFHz2WLAFC3TCOVjMzJEWawtxIc7oX9vziTF
1Uk2K4ccK2zdJlPI46fHjJrI2xXKZWkxVNkZ8LeMspckUqECQQDqhD0SoLXoRGks
h7byNZAMR5PfZTpHli7uFg9O+GoLtxQNE/rW6JPVcVkpCvs8oPPUu+1D7dHnyFiO
heyme35tAkEA1QEiny94KRtTuP/WEyyYUkRfltYjrAX1BC73Xu395cNwjvnNw7qI
f2dFUm5akGijk9UtL1qNxg+akBgJXkbkiQJAXbUHXkkfRrcHO4bjIDcs3us++BXP
yskE6Zeg+FIktZerCGrCYVs/rxsCoHbF2v0JUSjibrE5nZ8dW53B6OgRpQJBAKfr
9zFrqN0vT/eeqVQAai0g/gLZ2tF4+MpNhHLwSKNkSk5NHSxa19UowvvTR85kz+Bx
xOd6Ch7EmmNSr8AFP5ECQQDOXmjIecxNI51of9u6g4T2ITRcHTYyCqWLO6VqAWlD
G6ej+6/h+8DQyfJKMNbfMCGjZ7xZC3isNMmFibGQTLZD
-----END RSA PRIVATE KEY-----
-----BEGIN CERTIFICATE-----
MIIC7DCCAlWgAwIBAgIBADANBgkqhkiG9w0BAQQFADBUMQswCQYDVQQGEwJVUzEU
MBIGA1UECBMLTWlzc2lzc2lwcGkxDzANBgNVBAcTBkJpbG94aTENMAsGA1UEChME
SUVURjEPMA0GA1UECxMGU0lQIFdHMB4XDTA0MDkxMzEwMzg1NVoXDTA1MDkxMzEw
Mzg1NVowVDELMAkGA1UEBhMCVVMxFDASBgNVBAgTC01pc3Npc3NpcHBpMQ8wDQYD
VQQHEwZCaWxveGkxDTALBgNVBAoTBElFVEYxDzANBgNVBAsTBlNJUCBXRzCBnzAN
BgkqhkiG9w0BAQEFAAOBjQAwgYkCgYEAwyERDCCEvbwQRE9hRR7LNi28K4v5yvgD
KFGe+FE92phv22IA5x/OnlFBIxn1d3H1xKxV0pyPOrFAC6Co48BmAaIlLmjIcnq2
chfg/UrdqZZwXqun6q22j6S+m7yABet+vztvhhT9y0i4+ikuB0MJ7GlAa9mM8pAQ
9OT38Kx9vFUCAwEAAaOBzTCByjAdBgNVHQ4EFgQUlZRLaS3Zm/b0xWcq7TSnQMHM
7w8wfAYDVR0jBHUwc4AUlZRLaS3Zm/b0xWcq7TSnQMHM7w+hWKRWMFQxCzAJBgNV
BAYTAlVTMRQwEgYDVQQIEwtNaXNzaXNzaXBwaTEPMA0GA1UEBxMGQmlsb3hpMQ0w
CwYDVQQKEwRJRVRGMQ8wDQYDVQQLEwZTSVAgV0eCAQAwDAYDVR0TBAUwAwEB/zAd
BgNVHREEFjAUghJiaWxveGkuZXhhbXBsZS5vcmcwDQYJKoZIhvcNAQEEBQADgYEA
SufJHtereahZlkE5ssRRZRd/erLpEe2uUfHnTOydPBKOkvhVG4Vr4aoroPlE7gJK
a/2BF9bohwAUSC5j5q3nvuhUcoK9XZYm2nLkN3IAhCU6oswVBJAxLanGUCjR5sxS
HfGhGsqLmTEQ22HsrtLo68IYiwftXcLZbep50gRVX6c=
-----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 colons. 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:

```
kJl0ILrbzGtQX4zW4GlPo5DELq1hYXgfvI77xeQ1H7mXblNJBf6cLE0JAnRiDMp+
tbwSi9tj7JoknqeZAXtj5czqAKskj7axdYfe40basFy34HhNVc3WH2c3TwAlqbrm
kspEbEWUnBnIRXjnihQ3Pi5rHwUVkKPdogI26IqRgQE=
```

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:

```
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: "kJl0ILrbzGtQX4zW4GlPo5DELq1hYXgfvI77xeQ1H7mXblNJBf6cLE0JAnRiDMp
          +tbwSi9tj7JoknqeZAXtj5czqAKskj7axdYfe40basFy34HhNVc3WH2c3TwAlqbr
          mkspEbEWUnBnIRXjnihQ3Pi5rHwUVkKPdogI26IqRgQE="
Identity-Info: <https://biloxi.example.org/cert>;alg=rsa-sha1
Content-Length: 0
```

biloxi.example.org then forwards the request normally.

12. Identity and the TEL URI Scheme

Since many SIP applications provide a 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 (RFC3966 [12]). 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 [13] 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.

13. Privacy Considerations

The identity mechanism presented in this draft is compatible with the standard SIP practices for privacy described in RFC3323 [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 which the authentication service is willing to authorize, there is no reason why private SIP URIs which 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 'invalid.net' 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 their identity private". The use of the domain 'invalid.net' implies 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 RFC3323 requests that a privacy service to 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.

RFC3325 [10] 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 which 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 RFC3325, the mechanism described in this specification adds no information to SIP requests that has privacy implications.

14. Security Considerations

14.1 Handling of digest-string Elements

This document describes a mechanism which 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 RFC3261 for including headers in tunneled 'message/sip' MIME bodies (see Section 23 in particular).

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 replay 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 RFC3261 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). This 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 their own request (reusing the From, To, Contact, Date and Call-ID fields that appear in the original message), they 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 their 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 pre-empting 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 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 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 user trusts 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 RFC3261 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 their identity which 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 merely result in a failure at the verifier. Numerous changes that a man-in-the-middle might make would have the same effect. As such, 'alg' does not seem to introduce any new security considerations for this mechanism.

14.2 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 term discerning the

legitimate display-name for a user; it is safe to assume that impersonators will be capable of creating SIP accounts with arbitrarily 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 14.1.

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.

14.3 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 they receive a challenge via a mechanism such as Digest), then it is possible that a rogue server is attempting to pose as a 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 into 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 is 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.

14.4 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 which 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 RFC3263 [4] 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 their requests through a host like 'sip.example.com', and it may be that latter host which 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 RFC3263 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 by 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 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 as follows:

First, a verifier must acquire a list of one or more domain names which constitute the subject(s) of the certificate. A verifier MUST extract the subject CN field from the certificate. If the CN contains a domain name, it is added to a list we will call the 'subject list'. A verifier MUST also extract all subjectAltName fields from the certificate. If any subjectAltName fields contain domain names, these domain names should also be added to the subject list.

Once it accumulates the subject list, the verifier MUST compare each name in the subject list to the domain portion of the URI in the From header field of the request. If the domain portion of that URI matches any domain in the subject list, the verifier should consider the certificate to match the URI in the From header field for the purpose of verification.

If no member of the subject list matches the domain portion of the URI in the From header field, then the verifier should consider the certificate ineligible to sign the request.

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 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 pre-existing key exchange has justified such trust.

14.5 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 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.

15. 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.

15.1 Header Field Names

This document specifies two new SIP headers: Identity and Identity-Info. Their syntax is given in Section 10. These headers are defined by the following information, which is to be 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

15.2 428 'Use Identity Header' Response Code

This document registers a new SIP response code which is described in Section 7. 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 is to be 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

15.3 436 'Bad Identity-Info' Response Code

This document registers a new SIP response code which is described in Section 7. 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 is to be 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

15.4 437 'Unsupported Certificate' Response Code

This document registers a new SIP response code which is described in Section 7. 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 is to be 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

15.5 Identity-Info Parameters

This document requests that the IANA create 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 which 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.

15.6 Identity-Info Algorithm Parameter Values

This document requests that the IANA create 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 which 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.

16. References

16.1 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", 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., "Augmented BNF for Syntax Specifications: ABNF", RFC 2234, November 1997.

[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.

16.2 Informative References

[9] Kohl, J. and C. Neumann, "The Kerberos Network Authentication Service (V5)", RFC 1510, September 1993.

[10] 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.

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

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

[13] Faltstrom, P. and M. Mealling, "The E.164 to URI DDDS Application", RFC 3761, April 2004.

[14] Peterson, J., "Retargeting and Security in SIP: A Framework and Requirements", draft-peterson-sipping-retarget-00 (work in

progress), February 2005.

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

Appendix A. Acknowledgments

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 fixed to innumerable sections of the document. The bit-archive presented in Appendix B follows the pioneering example of Robert Sparks' torture-test draft.

Appendix B. Bit-exact archive of example messages

The following text block is an encoded, gzip compressed TAR archive of files that represent the transformations performed on the example messages discussed in Section 11. 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).cert: 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 --
H4sICOOGdEICA2lkZW50cmVmLnRhcgDsW02v41haLpBYYFEb/sBEIxagTFX87eT2
1GiOv78TfyfZINvxR2zHjj8SO6HEHwCh2aFBiA0LVixZgGYJG8SazWxYsOQfICGc
e6u6urv6dtdI3aXu6XskXyev7ZNz/J73ed73se9+F5VdE8WzZ99eg2Ecpghi3GMo
hZHjHsVxEhn3n7ZnCIxixPiHxLBnMIKRKPpsQjz7CO3Udn4zmTzLjlEXNW1VPnLe
IU+a6nR8UbzYPfstavu3/ve7wi87/2UYNd037X8Ehkkcf8z/KIJgo/8RFEEpBCWo
8XwSufkffvL/t95e3BrNCZI+YTjTlniJATZ3b4U0SWJmNsOAg5+AXqJBMm4s0Okk
r9N8Lyx6mAaGwQOW3mpG2zPGhnUNQ+B62XWunAFpABcA4nAMrYkm6p7Dw+64sTlL
o8G9nR4M2Vib7WZtSEFpppoB90x/34nC9aYMma4paKbTcw9GlQeDbcOFwYxDsVyQ
uHDYi2moa6zRL23uqtlg0K7g6t3bnB76nDH78mF+1SihDxnmV40SejvMZDvvWWMj
```

K9VWSs+hDgyOpg3AJhsYaJIgg0qggdpHm+twhu10g/LnE5IzCcS7itHIa7QFZhzF
Mn1i93yEUF4aOCKnUlk4gKgNosiujlfkOF+IR89LGtpPrYTulV0HaeVMs+TyiDa4
mlsD2CRkRu9NhTnClH/YLxNlJ7BLZsnMmXXmipuykmzPuAKC5WCVm+4g/ZJd1oOx
hY8t3IV7HEGMq9MHA0jGWwSELNkZfbLjevF2Y014SdMbjhdixqFCfagtBdJPzZmv
r/MkI47D/FSN86XpRHdFS+OijMYT2iz7YomxoZ/vs6CmBr+dSdLUD2O1VvgddOTb
3rM5VQP5w8JJNcZ1tYFjwfLW0bg0bRreFQEml77HDVwGjAd76DAOD0OBwMOMQMCj
X3UNtA/+TjXOclzHzDhXo7UH22BoDryQndF1DFcYEr8Tk4QD41LubyeMszR4DYzz
jue9mNzP16TpsOc3kr25rRFv/BGLKCKB70JhKNSDfg5soEPvBc1t5AYIYX9GpA55
qU176OuKPpn5bG/VbCmzMxo2jNLi1XqXXSDDna0OU784AZjwVZP1arKfneJ+jYor
dXleb07BsdSv2dH3mAFbqg0eEHpftrFuDZ1Cy9BZWqy8PrU80tWqWWijRZHqpx1z
0KfqerWbWeoFS0rhvO76sPGAPdfczSFnHMM2z4Hnm1BsvHoF3UMDp7Pvw8Wzp/Zb
wP/HZn/Oo8tH5H+EIKkv8j9JPfH/x+Z/0wKTlSm5Y0BPFG7zLgcYaW/EeWWEK2Yu
HrR5sOBwT08DqtuCinatXLm4iwZwa6y8GEGCp+suQioPoQdmCuGYZoiNgJU5qYQL
P16ZpFtUSu9VTgVCpgxseXbl9ltyV08ZgmsNQVpW/Sa3knq3RI8VlFVMaSbXJEvO
oMARE5VJhkNyrV8ag84Mq9IOq3kxd4SdEujqWjrpO0dDNGFuSCwwAA2NzApAt1oK
wL3o1XRtLeWBm6IOLfq15qnGRQQKxVSXlqD46bI8xVUo14I9ZhQpr222S6iIjBRW
sjQJ+6WjVVRCd6eq8wwnCFTRFpT1njbJVR0cSOac9rJ10jdn+LzUT3aAWFpRQIq/
S3Cs9HF6kRvRpSuRC1nEuZ0raLWpc9bdogAcVEtt4rRAHMYw2CPl8lKXHMpL0CvQ
Sunldp6U/PQ0VRbZtPXDGXZOhKRZno+DW/eVti7IyNNns+N2FtZt6peseuiahZ9m
HsNC7LBxL3nRgJwD1x0pqzJIhXm4vRxcxiLkGOYpHrjDcYDpxFyJvVzsL05CLvFs
tZkCK4FCdUWW14WPzXtva2RkY16SOBa24jqg+cN0PiZTNJAPWy4mOMtSyCnh7jS7
0OtTAHaZ3EM+rTkbxIGnpa+SJLjEG8+R6mnGSvOpGau5YsxFWIrP0XCucg+14t5a
KUh+DWOOMdjZEtIMDx2SvpvbaXqIFMZAZkSMennRmkxBrISLOPXZqDYuUpIt/VXB
XO2MQyRsKh8du7mYUL+Y2Tt85Nmma5gzQuvs2LPBJKJ05ecq30kjPvULDXCgYi5s
0In8issoJJjWAopuQhyyVjwNggqTheX00NCwtp4JvdxMMTZOrproO8Os25erZnpC
2Hc8+aVh9R3G/1PwDcP/1+E/jKO3+h/BcITAMQK/4T9OEk/4/5Hxf+XQqsR8FvqF
WAOwwFi1YEkBxo4VCzAcMJZLOmAZem98NSVAj3HCh1IC9BgnfCglQI9ywqfR+cVJ
/2Dzv2BfVMP+25B/vlb/gUn0bf6Ho9Qt/yMoCn+K/++C/kOxDAMK7+v0H+dLhBUH
0mjprbKi2l5xDdH7rQ+FfGCvt47uS3SbzuVjICxw/wvVOfSmPF/dYOihPNcEB34o
yzUaX7M2gDU2H7Qr12vXBNHd6mZD3tqgN8beZd8XD3gWWG/FAwZGjiGmP2wifdSM
e7WGhcb5iFy/ZXxvOEe3YdtAfbiosmmu4F1u85mptDZd6LLD0GvzytDl9b26n7uv
+zMD9MkmZ5INB/oLZ473mDsHvWFyi9Q0KVXfo3MFPxOXc8JCCi9EU55boMf0jKIS
IIbZsix4WhpKZIeJyKAMLny8rJYNDxiSqfA5fQC+VKiHTArLGoXCNE5mTrOrt9t+
XZ9KskbRjLSmB+oC6Kibnq/dOU3txQXe49N9fhoTG5kSCuAvRmA/AgNaLG1srgyL
M+8woOcA8Jf01WboSwZ29zKOgXN8YjjF1lR9C9seZgE8eGFN2VZpaKIGUf28j8G9
WJLRotOHOHj0ZKqfpp5iehpvDMwVyLcfgGiwsUHh2ppp9Fxyv8okru90f61fHza6
HxfPp+uEHteJcSjaAEuPN7UOek+ue+Pfm1w3utd+UOkiZiSHnn0YqU0Dpx+nS8+u
YAfdz9PkOD4DTpLK+zfr4bRdp2mwptutRZzDQ/hFiY97kPg4AFmnWBZHhIn8dFvk
HNG2prk1d7OoUY9chJ6cWCzt5WW3opVlfk5dAXcb3K+aalVwVCIrkD9DaX4RVGkP
HIshMqLGyvMpdcJKWay3mwNaqrmOSSBlHLJqe5eWwaD6peAwmUm0gwWJsZAKba0e
bM5AUbFtOrUi59Jm38fdOlS3QXQk4MR012T4JPP8UPn/W5B/fhP95w3/kwSOPPH/
d0//YVnJ5EaTtTjTNGejPC+2LaoGfbOflcoUaJWzpXDHxvyDMAs2CVtC4pyMHIfL
hNjFwnjNNfzaKqU52TpArZQs64UNXe2dk6+ll6jZXvgpu7A69FiUdFSTcQ01gb86
qmRwlgBPYuRsiU4FfoWplpqRioWLRl+2vkM36OZ6yWkWJlZx34jo3P2s/rNVTZnv
k8ILlc32mOkKgduEuAPCCtn2FXoVsCzceKhjb2dccvK8gBKvbaDHp+31CAlemIj9
acmh83IjGgmjAJT0z/FSOEUBL15RTwU8g9nM0s20q8x5ft8NUkhV68X5urd5CHFy
VMHDUEGvO7lYSTgZi5ncSOiwVrZePrh6vp2rkdYew9ypb6oEW6csbFXqujKFvIVS
KrjoW6CZ4+S29lEs9tSJTxbLqVCp3WDo3KzxSHnlhm5+ZM7tvFqtnNMUYamdWF74

/RJKo8shwojupv8gBrcvLwtcMTv7tJp53OWycXIzLrpN1oA1QjMUtj5hCyLU++xc
6j1VS1CM7njnQPi5sM/yhdOpSK0PydTP6URe50G+N2SwDhxxneex2YTiEg8yiQ1b
7NROp/R6BV3anCO3UTLlpbzbRg0jNMzGbWfN0DKVGPDoGZYdK9sHDUeU2/nOIzCa
XCbm0ZBpoMQNtLjyTa3DZ3sWRbVrAH8PJ7NE3aIdj0+1o56Kam8pem7lhC5ag48s
nKo/n21zTuTXKT1Aw3JHMinFHQ661cwBvyLu7/RyPSZKUTjoEoFU8eJEJriNSrYZ
ivbmwtSeuiTdGngFCwlklE3JWTqds8YllhVND2KNEbItNWwZbN/q2oHfB4Jhq9vv
n/7zFv+/efnna+s/An5T/+EoSWDEDf8x7An/vx/6z+OUAD3GCR9KCdBjnPChlAA9
ygmP6j9P9fBTPfxUDz/Vwz+Uevgd/1+il6FfVuU+9IuPyv+31/3u3/9EYQQjMPjG
/yhGPfH/x2jt/ngXVMHP36R/0eAfjkX0smqS17dDfrEPo5+/fTb49mhYHV77czzA
Q4qMSJJC4NcohkzoDffaTk8/maDIhI+CCTo6eoLgd8jijkAmgma/fv38ST76Dsf/
/Zd99w2n/1+r/yAo8e79b/Q+/hH86fnPR2ljzE7eBfqY72EvvyTaJ5a0mqEv4eeQ
u/fv3n6b2ao1QRbjp5foS/yToPHLMH11XaQCHiil36a7Fhkv0fzhBV81vd/s2rsJ
NVr4pjrcTegqmPz0cfz52Sedn7zyyXBOhPFzyK7uJuA2yIdrHgWmh8vGUc0xGKEo
/DnE+l10N/lKYHoOMX5RvJDYu8nncW08YEX13eQNvD2HpDchcjf5cS4XsKQ2wVXo
jDV+9XChWFXEmFjXSLpZJ/FZoqghMhCROqyDMQ2mYzJUOVgGpblnteNzaPJpm3ZB
b+0XXUbJVV7W0Rasu4wIrzVQ2jyj/GG3iSMcDvyWv2C4mOpuiHkiGmJ2D4o6aD7b
1yFvj1zAeU5Jl5K5zsp9amCrPdGIvePmympXJRJKSrWZGNyrH0/eTemFVMbjTf5p
2nXH9m42e98ls9vT4Z994hfJq6b1X7Spj4w3qCq7sYMXalQmXXo3gZ8w/vuK/4eo
bf0k+sjP/2F8tN30HxInSfR2HkJQ5JP+84T/3wz+fyi0/yCB7HPxfwP0b+E3vi7+
YYp8V//d7AiBEE/v/32UZokA+ePP1f5/8mqCzefwDiZQzA99dEdR6HwRhNRuvtjF
ZDy/hdEOwdEQfyL637L43ydltPvo8Y+++f/fW/yT989/COKJ/z9O+8Uvu8m//Po/
8tWf/uV//td++fe/+NeX/+j7w49+9Vf/+2//Tfzon/+6tKTf+7+//Yn2+7/bBX/w
73/+T7/6w7/4dfg/f/NHf/cPv/ydP/v/9s0ktIkoDuNasOJgQUFBkdJB9FACzeyT
BKfQdE0hbbRJbEN7mEymySSdZJrNJIjQU6UILrihniriBi4IWnsRPYjrSfQgVITi
LijoQUWok5Ta1DYZQs1Q5f8jh4GQIXnJ9733vu9FePXo/bKLW/Zfv5oYfjHZ8vxK
742XKbvr8bMaP9u5bOv58deHHjQNNB8+tfFec+rNyDq2vi5Us2L9gdMToxXM25GD
Q8tBgUtO/3ViWNBX/zT7W/8sRk73vzD/68IiQwwDssjkQkZKiStgwVE2/UvhpBQv
TwWkpX8aI2f0TzBETv8sg4H+9aB4yVOkHZq7mbaQOIXTZtTW4bY5F2qBSAtGWDAy
1wJp5w27k5y6+45wrpgYbUDV7QdmoiiaYPIvbR2ozUGhhW5RhcS4LjEWkyJhtKvJ
UYUInOYr4hyW3fXLHJ/wSRGUMuMsge5wOowNbkf2CZ6LxhWZVywY6mi0u4wm9cf7
j4cEf+q/HBWQ5vnf/P6HJOD/3zoyLVh0ntBndF4k+SukowWDQNN0gqeV+c1kg2VJ
+uZ41EK5ZKGmKM+8pmNCXohb8t9doaGon1MYdbT7Ew1eRe5RVzkDqRZ5oDOUkO02
g93THbQSktvYsZNOxVq2D2aajAqVliLJaLt1MOAjnZ6Q6A/nlzwxOdIWTWdCZr/T
2sa2Bo1iVEo1t9j6TUSEbAjLXkOjzRv1+wYw0t/KNzG2xmRIcQ9G7N0eT9CX7Mq/
l5IW0m2dgQAuyK4U6TW7o6TiSTQHDIIVwxWvfbvJigsBg0GyB3dxmwv3RQt8/qKF
kTOtqKPNK4o6hHxc9WljzKfMj2Fxiq1CqhCYEXTx/3JUQJr9P0bO9j9sbv1Hgf+D
//8X/g9uB/xD/l+OClDz/Cc2e/4Tx7Prf5qhIf/XhVz/92f2U8upRiOaWYESVXMl
cbP65fBivzop46QgUCTP8OrDywgYC+r/z/RfhgqwlP4ve509/83A+U9doI7XLR/v
51Nvjq3tP/yu7videvzI5MrAkZ9Pb9697biITZ3dNzr6437lub4JxpdZ9fHE0IeH
Zm4Ia/0YmMqMXV2zo+ZLkK1In/y2edOlleKTVGWgfi/yvdZ9YXXfUSH59fz9R8OW
DYmoiE/ccl3O9Iy17VGq6TPX1nyu7h3+NEpsAwUuRf3/3QqwlP6PzPV/LEZD/q8L
i8ykYsgigygFKSV9gtUGAAAAAAAAAAAAAAAAAAAAACAFr8AHiHENgB4AAA=
-- END MESSAGE ARCHIVE --

Appendix C. Changelog

NOTE TO THE RFC-EDITOR: Please remove this section prior to

publication as an RFC.

Changes from draft-ietf-sip-identity-04:
- Changed the delimiter of the digest-string from ":" to "|"
- Removed support for the SIPS URI scheme from the Identity-Info header
- Made the Identity-Info header extensible; added an Identity-Info header for algorithm with an initial defined value of 'rsa-sha1'
- Broke up the Security Considerations into smaller chunks for organizational reasons; expanded discussion of most issues
- Added some guidelines for authentication service certificate rollover and lifecycle management (also now based HTTP certificate retrieval on RFC2585)

Changes from draft-ietf-sip-identity-03:
- Softened requirement for TLS and direct connections; now SHOULD-strength, SIPS and Digest auth-int listed as alternatives.
- Added non-normative section about authentication service behavior for backwards-direction requests within a dialog
- Added support for CID URI in Identity Info
- Added new response codes (436 and 437) corresponding to error cases for an unsupported URI scheme and an unsupported certificate, respectively

Changes from draft-ietf-sip-identity-02:
- Extracted text relating to providing identity in SIP responses; this text will appear in a separate draft
- Added compliance testing/example section
- Added CSeq to the signature of the Identity header to prevent a specific cut-and-paste attack; also added addr-spec of the To header to the signature of the Identity header for similar reasons
- Added text about why neither Via headers nor display-names are protected by this mechanism
- Added bit-exact reference files for compliance testing
- Added privacy considerations

Changes from draft-ietf-sip-identity-01:
- Completely changed underlying mechanism - instead of using an AIB, the mechanism now recommends the use of the Identity header and Identity-Info header
- Numerous other changes resulting from the above
- Various other editorial corrections

Changes from draft-peterson-sip-identity-01:
- Split off child draft-ietf-sip-authid-body-00 for defining of the AIB

- Clarified scope in introduction
- Removed a lot of text that was redundant with RFC3261 (especially about authentication practices)
- Added mention of content indirection mechanism for adding token to requests and responses
- Improved Security Considerations (added piece about credential strength)

Changes from draft-peterson-sip-identity-00:

- Added a section on authenticated identities in responses
- Removed hostname convention for authentication services
- Added text about using 'message/sip' or 'message/sipfrag' in authenticated identity bodies, also RECOMMENDED a few more headers in sipfrags to increase reference integrity
- Various other editorial corrections

Intellectual Property Statement

Disclaimer of Validity

Copyright Statement

Acknowledgment

Funding for the RFC Editor function is currently provided by the Internet Society.