SIP WG J. Peterson
Internet-Draft NeuStar
Expires: August 17, 2005 C. Jennings
Cisco Systems
February 16, 2005

Enhancements for Authenticated Identity Management in the Session Initiation Protocol (SIP)
draft-ietf-sip-identity-04

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 recommends practices and conventions for identifying end users in SIP messages, and proposes a way to distribute cryptographically-secure authenticated identities.

Table of Contents

1. Introduction 3
2. Terminology 3
3. Background 3
4. Requirements 6
5. Overview of Operations 6
6. Authentication Service Behavior 7
6.1 Identity within a Dialog and Retargeting 9
7. Verifying Identity 11
8. User Agent Behavior 12
9. Proxy Server Behavior 12
10. Header Syntax 13
11. Compliance Tests and Examples 15
11.1 Identity-Info with a Singlepart MIME body 16
11.2 Identity for a Request with no MIME body or Contact . . 18
12. Identity and the TEL URI Scheme 21
13. Privacy Considerations 22
14. Security Considerations 23
15. IANA Considerations 27
15.1 Header Field Names 27
15.2 428 'Use Identity Header' Response Code 27
15.3 436 'Bad Identity-Info' Response Code 28
15.4 437 'Unsupported Certificate' Response Code 28
Authors' Addresses 29
A. Acknowledgments 30
B. Bit-exact archive of example messages 30
B.1 Encoded Reference Files 31
16. References 28
16.1 Normative References 28
16.2 Informative References 29
C. Changelog 33
Intellectual Property and Copyright Statements 35

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 canonical SIP address-of-record URI employed to reach a user (such as 'sip:alice@atlanta.example.com').

RFC3261 stipulates several places within a SIP request that 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 accurately, 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 TLS or S/MIME, 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 begin sending presence notifications. 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 be 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 SIP user agents support a means of authenticating themselves to a SIP registrar, commonly with a shared secret; Digest authentication, which MUST be supported by SIP user agents, is typically used for this purpose. 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 of requests in order to provide a 'return address' identity to recipients. The identity mechanism specified in this document derives from the following principle: if you can prove you are eligible to register in a domain under a particular address-of-record, you are proving that you are capable of legitimately receiving 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.

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 [8]. 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 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 act as an authentication service. 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:

- o The mechanism must allow a UAC to provide a strong cryptographic identity assurance in a request that can be verified by a proxy server or UAS.
- o User agents that receive identity assurances must be able to validate these assurances without performing any network lookup.
- o User agents that hold certificates on behalf of their user must be capable of adding this identity assurance to requests.
- o 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.
- o The mechanism must prevent replay of the identity assurance by an attacker.
- o 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).
- o It must be possible for a user to have multiple AoRs (i.e. accounts or aliases) under which it is known at 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.
- o It must be possible, in cases where a request has been retargeted to a different AoR than the one designated in the To header field, for the UAC to ascertain the AoR to which the request has been sent.

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 populate the value of the From header field (which 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 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 authenticate 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.

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 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 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. 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.

The authentication service authenticates the identity of the message sender and validates that the identity given in the message can legitimately be asserted by the sender. Then it computes a signature over the canonical form of several headers and all the bodies, and inserts this signature into the message.

First, an 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.

Second, the authentication service needs to 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).
- 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 authorization 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'. 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 is anonymous (per RFC3323 [3]), then the 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'). Some SIP user agents that receive requests render the display-name of the caller as the identity of the caller. However, there are many environments in which legislating the display-name isn't feasible, judging from experience with email, where users frequent make slight textual changes to their display-names. Ultimately, there is more value in focusing on the SIP address of the sender (which has some meaning in the network and provides a chain of accountability) than trying to constrain how the display-name is set. As such, authentication services MAY check 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 'Inappropriate Display-Name' response code. However, in many environments this will not make sense. For more information on rendering identity in a user interface, see Section 8.

Third, 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 are provided in section Section 10.

Finally, the authentication service MUST forward the message normally.

6.1 Identity within a Dialog and Retargeting

Retargeting, the alteration by intermediaries of the Request-URI of a SIP request, 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.

If there were a means in backwards-direction requests to signify a 'connected party', an identity of the unanticipated user whose SIP endpoint was reached by the dialog-forming request, it isn't clear that it would actually be beneficial to provide a corresponding Identity header signature over that information. The Identity header is designed to prevent impersonation-based attacks, and it is very unclear how and why an attacker might attempt to impersonate an unanticipated third party in a backwards-direction request within an existing dialog. That is, it's unclear how the caller's potential authorization policies would be any more successful at thwarting impersonation if new requests in the backwards direction came from an assured unanticipated third-party instead of an unassured unanticipated third-party. Thwarting impersonation is, ultimately, the purpose of this Identity mechanism, and it must be left to other mechanisms to solve other security problems for SIP.

The mechanism in this draft cannot aid in determining whether or not the unanticipated party is an appropriate target of this request and, accordingly, solving this problem is outside the scope of this draft. If, however, it were possible for the sender of the dialog-forming request to anticipate that retargeting had occurred, and to gain some kind of assurance of the new target of the request before any requests in the backwards direction were received, this would open up some new approaches to authorization policy.

Any such 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 guidandance 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.

7. Verifying Identity

When a user agent or proxy server receives a SIP message containing an Identity header, it may inspect the signature to verify the identity of the sender of the message. If an Identity header is not present in a request, and one is required by local policy (for example, based on a global policy, 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, the user agent or proxy server MUST first 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 retained 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), the verifier MUST send a 437 'Unsupported Certificate' response.

Subsequently, the recipient MUST verify the signature in the Identity header, and compare the identity of the signer (the subjectAltName of the certificate) with the domain portion of the URI in the From header field of the request as described in Section 14. Additionally, the Date, Contact and Call-ID headers MUST be analyzed in the manner described in Section 14; recipients that wish to verify Identity signatures MUST support all of the operations described there.

If a verifier determines that the signature on the message does not correspond to the text of the message, then a 428 'Invalid Identity Header' response MUST be returned.

Once the identity of the sender of a request has been ascertained, various policies MAY be used to make authorization decisions about accepting communications and the like. Such policies are outside the scope of this document.

8. User Agent Behavior

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 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 header field that asserts its identity (for a SIP request, this is the From header field). In a request it MUST set the URI portion of its From header to match a SIP, SIPS or TEL URI AoR under which the UAC can register (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).

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. UAC implementations MUST provide a way of provisioning pre-loaded Route headers in order for this mechanism to work for mid-call requests in the backwards direction of a dialog.

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. Proxy Server Behavior

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 verify the existence of an Identity 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.

For the purposes of identifying mid-dialog requests, proxy servers that instantiate the authentication service role MUST Record-Route themselves in dialog-forming requests.

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.

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

Identity-Info = "Identity-Info" HCOLON ident-info
ident-info = LAQUOT absoluteURI RAQUOT
```

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 colon:

- o The AoR of the UA sending the message, or the 'identity field'. For a request, this is the addr-spec from the From header field.
- o The addr-spec component of the To header field, which is the AoR to which the request is being sent.
- o The callid from Call-Id header field.
- o 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.
- o 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. The first letter is upper case and the rest of the letters are lower case. All requests that use the Identity mechanism MUST contain a Date header.
- o 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 colons in the canonical string).
- o The body content of the message with the bits exactly as they are in the Message (in the ABNF for SIP, the message-body). 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 colon 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, as follows: compute the results of signing this string with sha1WithRSAEncryption as described in RFC 3370 and base64 encode the results as specified in RFC 3548. Put the result in the Identity header.

Note on this choice: Assuming a 1024 bit RSA key, 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 Identity-Info header MUST contain either an HTTPS URI or a SIPS URI. If it contains an HTTPS URI, the URI must dereference to a resource that contains a single MIME body containing the certificate of the authentication service. If it is a SIPS URI, then the authentication service intends for a user agent that wishes to fetch the certificate to form a TLS connection to that URI, acquire the certificate during normal TLS negotiation, and close the connection.

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

Note, in the table above, that this mechanism does not protect the REGISTER method or 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. The REGISTER method uses Contact header fields in very unusual ways that complicate its applicability to this mechanism. Accordingly, the Identity and Identity-Info header MUST NOT appear in REGISTER or 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. Implementations MUST verify their compliance with these examples, i.e.:

- o 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.
- o 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 appear here only for illustrative purposes. Therefore, in compliance testing, implementations of verifiers SHOULD generated appropriate warnings about the use of self-signed certificates.

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:

CyI4+nAkHrH3ntmaxgr01TMxTmtjP7MASwliNRdupRI1vpkXRvZXx1ja9k0nB2sN
3W+v1PDsy32MaqZi0M5WfEkXxbgTnPYW0jIoK8HMyY1VT7egt0kk4XrKFCHYWGCl
sM9CG4hq+YJZTMaSROoMUBhikVIjnQ8ykeD6UXNOyfI=

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:
"CyI4+nAkHrH3ntmaxgr01TMxTmtjP7MASwliNRdupRI1vpkXRvZXx1ja9k0nB2s

N3W+v1PDsy32MaqZi0M5WfEkXxbgTnPYW0jIoK8HMyY1VT7egt0kk4XrKFCHYWGC
                   lsM9CG4hq+YJZTMaSROoMUBhikVIjnQ8ykeD6UXNOyfI="
      Identity-Info: https://atlanta.example.com/cert
      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 calcuating 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:

A5oh1tSWpbmXTyXJDhaCiHjT2xR2PAwBroi5Y8tdJ+CL3ziY72N3Y+lP8eoiXlrZ
Ouwb0DicF9GGxA5vw2mCTUxc0XG0KJOhpBnzoXnuPNAZdcZEWsVOQAKj/ERsYR9B
fxNPazWmJZjGmDoFDbUNamJRjiEPOKn13uAZIcuf9zM=

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: "A5oh1tSWpbmXTyXJDhaCiHjT2xR2PAwBroi5Y8tdJ+CL3ziY72N3Y+lP8eoiXlr
          ZOuwb0DicF9GGxA5vw2mCTUxc0XG0KJOhpBnzoXnuPNAZdcZEWsVOQAKj/ERsYR9
          BfxNPazWmJZjGmDoFDbUNamJRjiEPOKn13uAZIcuf9zM="
Identity-Info: https://biloxi.example.org/cert
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'). That username conforms to the syntax of the TEL URI scheme (RFC2806bis [9]). 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 MUST put telephone numbers in the From header field into SIP or SIPS URIs, if 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 in accordance with its local policies. Verifiers SHOULD NOT accept

signatures over From header TEL URIs in the absence of some pre-provisioned relationship with the signing domain that authorizes this usage of TEL URIs.

The guidance in the paragraph above is largely provided for forward compatibility. In the longer-term, it is possible that ENUM [10] 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 (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' may not be possible for privacy services that also act as authentication services. The assurance offered by this combination service is "this is a known user in my domain that I have authenticated, but I am keeping their identity private".

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 [8] 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.

14. Security Considerations

This document describes a mechanism which provides a signature over the Contact, Date, Call-ID, CSeq, To, and From header fields of SIP messages. 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 servers 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). Accordingly, if an Identity header is replayed within the Date interval, receivers will recognize that it is invalid because of a Call-ID duplication; if an Identity header is replayed after the Date interval, receivers 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 device 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 accordingly 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.

Note that this mechanism does not provide any protection for the display-name portion of the From header field, and thus users are free to use any display-name of their choosing, and attackers could conceivably alter the display-names in a request with impunity. If an administrative domain wants to control the display-names selected by users, they could do so with policies outside the scope of this document (for example, their authentication service could reject requests from valid users that contain an improper display-name in the From header field). While there are conceivably attacks that an adversary could mount against SIP systems that rely too heavily on the display-name in their user interface, this argues for intelligent interface design, not changes to the protocol.

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 agent that originated a SIP request if a man-in-the-middle can change the SDP and direct media to an different IP address. 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 man-in-the-middle will interfere with sessions, 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."

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 likely that applications building on the Identity header could leverage this integrity protection, especially body integrity, to provide further security services.

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.

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 uses some trustworthy practice to authenticate its users. However, it is possible that some domains will implement policies that effectively make users unaccountable (such as accepting 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.

Since a domain certificate is used by an authentication service (rather than individual certificates for each identity), certain problems can arise with name subordination. For example, if an authentication service holds a common certificate for the hostname 'sip.atlanta.example.com', can it legitimately sign a token containing an identity of 'sip:alice@atlanta.example.com'? It is difficult for the recipient of a request to ascertain whether or not 'sip.atlanta.example.com' is authoritative for the 'atlanta.example.com' domain unless the recipient has some foreknowledge of the administration of 'atlanta.example.com'. Therefore, it is RECOMMENDED that UASs receiving signed requests notify end users if there is ANY discrepancy between the subjectAltName of the signer's certificate and the host portion of the identity within the From header field. If the domain name in the subject of the certificate is subordinate to the domain name in the identity URI, then verifiers may consider this a minor discrepancy. Additionally, there are ways that a verifier might leverage the information about canonical SIP servers within a domain stored in the DNS (see RFC3263 [4]) to determine whether or not a particular authentication service is authoritative for a domain; however, this is a subject for future work.

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.

Finally, 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. Accordingly, these headers are only truly efficacious if the would-be verifier knows that they must be included in a request. In the long term, some sort of identity mechanism along these lines must become mandatory-to-use for the SIP protocol; that is the only way to guarantee that this protection can always be expected. In the interim, however, identity reception policies at a domain level or an address-book level should be used by verifiers to determine whether or not identity is expected from a particular source of SIP requests. Those authorization policies are outside the scope of this document.

15. IANA Considerations

This document requests changes to the header and response-code sub-registries of the SIP parameters IANA registry.

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: (none)
```

15.2 428 'Use Identity Header' Response Code

This document registers a new SIP response code which is described in

Section 7. It is used when a verifier received a SIP request that lacks an Identity header as a response indicating 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

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.

16.2 Informative References

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

[8] 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.

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

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

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, and Aki Niemi for their comments. 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:

- o (foo).message: the original message
- o (foo).canonical: the canonical string constructed from that message
- o (foo).sha1: the SHA1 hash of the canonical string (hexadecimal)
- o (foo).signed: the RSA-signed SHA1 hash of the canonical string (binary)
- o (foo).signed.enc: the base64 encoding of the RSA-signed SHA1 hash of the canonical string as it would appear in the request
- o (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:

- o (foo).cert: the certificate of the domain
- o (foo).privkey: the private key of the domain
- o (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 --
H4sICPGlE0ICA2lkZW50cmVmLnRhcgDsW8uv43hWLsTOomaFYMEmtIQAhar47eTO
FJqf3+/E78QakGzHb8dO7CR2spwFGok9IMSWLftZwwaxYQRCQixgC0JC/Ac491ZV
P6pvVyF1l3q675EsJyd+/F7n+8757OTbuD62cTJ79s0ZDOMwRRDjHkMpjBz3KI6T
yLh/a88QGIVxCkVJFHsGIxiJEs8mxLOPYKfuGLSTybNiHx/jtmvqR47blWnbnPYv
qhfbZ98hy9/Mf3CsgvoYvIzi9vh1zz8CwySOPzb/KIJg4/wjKIJSCEpQ4/EkgqLP
JvDT/H/j9uJmNCdI+oThTFviJQbY3L0X0iSJmdkMA3ZBCnqJBum4sUCn0/KQlbmw
6GEaGAYPWNrXjK5njA3rGobA9bLrXDkD0gAuAMThGFoTTdQ9R7vtfmNzlkaDez89
GLKxNrvN2pDC2sw0A+6Z/v4iCtebMmS6pqCZTs89OFUeDLYNVwYzNsVyQerCUS9m
ka6xRr+0uatmg0G7gqt373N66HPO4sub+VWthD6kmV/VSuhNM1N/3rPGRlYaX8rO
kQ4MjqYNwKYbGGiSIINGoIHax5vrcIbtbIPy5xNSMinEu4rRymu0A2YSJzJ9YnM+
RigvCx2RU6kiGkDchXFsN/srsp8vxL3npS0dZFZK98r2CGn1TLPkeo+2uFpaA9ik
ZEHnpsLsYSrY5ctU2Qrsklkyc2ZduOKmbiTbM66AYDlY5aZbSL8Ul/Vg+PC+g49R
jiOIcXX6cADpOERAKNKt0adbrhdvA2vCS5recLyQMA4V6cPBUiD91J75w3WeFsR+
mJ+asb80nequaGlcXNB4Spt1Xy0xNgrKvAgP1BB0M0maBlGiHhR+C+35rvdsTtVA
+bBwMo1xXW3gWLC8XWhcmjYNb6sQk+vA4wauAMaDP3IYh4ehUOBhRiDgcV51DXQP
851pnOW4jllwrkZrD77B0Bx4ITvj1DFcZUj8VkxTDoxLub8dMPbS4DUw9juZ92J6
31+TpqOe30j25rZGvPEmFlHFAn+MhKFSd/o5tIEOvRM0t5YbIIKDGZE55OVg2kN/
aOiTWc5y68DWMjujYcOoLV49bIsLZLiz1W4aVCcAE4Fqst6B7GenpF+j4kpdnteb
U7iv9WuxDzxmwJZqi4eE3tddolvDUaFl6CwtVl6fWR7pas0sstGqyvTTltnpU3W9
2s4s9YKltXBeH/uo9YA919zNrmQcwzbPoReYUGK8egXdQwOns+/CxbMn+w7w/77N
z2V8+Yj8jxAk9UX+J6kn/v/Y/G9aYLIyJXcM6InCbT7NAUbaG3FeGeGKmYs7bR4u
ONzTs5A6+qChXatULu6iBdwaqy9GmOLZ+hgjjYfQAzOFcEwzxFbA6pJUokWQrEzS
rRql9xqnARFTh7Y8u3K5T24PU4bgOkOQlk2/Ka30sF2i+wYqGqY202tapGdQ4YiJ
yiTDIaXWL41BZ4ZVbUfNvJo7wlYJdXUtnfStoyGaMDckFhiAhkZmBeC4WgrAvejN
dG0t5YGbog4tBgfNU42LCBSKaS4dQfHTZX1Kmkg+CPaYUWS8tvGXUBUbGawUWRr1
S0drqJQ+npqjZzhhqIq2oKxz2iRXh3BHMuesl62TvjnD56V+skPE0qoKUoJtimN1
gNOL0ogvxxq5kFVS2qWCNptDybo+CsBOtdQ2ySrEYQyD3VMuLx3TXX0JewVaKb3c
zdOan56myqKYdkE0w86pkLbL835wD32jrSsy9vTZbO/PokOXBTWr7o7tIsgKj2Eh
dti4l7JqQcmB65aUVRlkwjzyLzuXsQg5gXmKB+6wH2A6NVdiL1f5xUnJJV6sNlNg
pVCkrsj6ugiwee/5RkG25iVNEsEX1yHN76bzMZmigbzzuYTgLEshp4S71exKP5xC
sC3kHgpozdkgDjytA5UkwSXZeI50mBasNJ+aiVoqxlyEpeQcD+em9FAr6a2VgpTX
KOEYg50tIc3w0CHtj3M7y3axwhjIjEhQr6w6k6mIlXARpwEbH4yLlBbLYFUxV7vg
EAmbynvHbi8m1C9m9hYfebY9tswZoXV2vLLBpKJ05ecqf5RGfOoXGuBAw1zY8Cjy
K66gkHB6EFB0E+GQteJpEDaYLCynu5aGtfVM6OV2irFJetXEwBlmx7xetdMTwn7K
k18aVt9i/D+FXzP8vw//YRy91f8IhiMEjhH4Df9xknjC/4+M/yuHViXms9AvJBqA
BcY6CJYUYuxYsQDDAWO5pAOWoXPjqykBeowTPpQSoMc44UMpAXqUE95G5xc7/b3N
/8K8aob8m5B/3qv/wCT6Jv/DUeqW/xEUhT/F/7dB/6FYhgGV9z79x/kSYcWBNFp6
o6yotlddI/R+6yOhHNjr7UL3JbpNl/I+FBZ48IXqHHpdnq9uMPRQnmuCAz+U5RqN
r1kbwBpbDtqV67Vriuhuc/Mhb3zQa2fvsu+KBzwLrDfiAQMj+wjTHzaR3mvGvVrD

QmN/RK73mcAbzvGt2TZQH05qbJqreJfbfKYrnU1Xuuww9Nq8MnR9fafu5+7r/sIA
fbopmXTDgf7CmeMYc+ewN0xukZkmpeo5OlfwM3E5pyyk8EI85bkFus/OKCoBYpgt
64qnpaFGtpiIDMrgwvvLatnygCGZBp/TOxBIlborpKg+oFCUJenMabcH3+/Xh1NN
HlC0IK3pjroAOj5Oz9fjOcvsxQXO8WlensbERqaECgSLEdj3wIAWSxubK8PizDsM
6DkAgiV9tRn6UoDtvYxj4ByfGk7lm2pgYf5uFsKDFx0o26oNTdQgqp/3CbgXSwpa
dPoIB48eTPXTzFNMT+ONgbkC+XYDiAYbG1SurZlGz6X3q0zi+qMerPXrw0b34+J5
u07ocZ0Yu6oLsWx/U+ugd+S61/N7k+vG6bUfVLqYGcmhZx9aatPA6cfu0rMr2EL3
/TQ5ji+Ak2Zy/no9nPx1loVruvMt4hztoi9KfNyDxMcByDolsjgiTBxkflVyRNeZ
pm9uZ3Gr7rkYPTmJWNvLy3ZFK8vynLkC7rZ40LTNquKoVFagYIbS/CJssh44FkMU
xAGrz6fMiRplsfY3O7RWSx2TQMY4ZNP1Li2DQQ1qwWEKk+gGCxITIRO6g7qzOQNF
xa49qg05lzZ5nxzXkeqH8Z6AU9Ndk9GTzPN95f9vQP75/+g/r/mfJHDkif+/ffoP
y0omN7qsxZmmORvlebHrUDXs23xWK1OgNY5P4Y6NBTthFm5StobEORk7DlcIiYtF
yZpr+bVVS3Oyc4DaKEXRCxu6yZ1ToGWXuPUv/JRdWEd0X9V0fCCTA9SGwWqvkuFZ
AjyJkbMlOhX4FaZaakEqFi4afd0FDt2im+ulpFmYWCV9K6Jz97P6j6+aMt+nlRcp
G39f6AqB24S4BcIK8fsGvQpYEW081LH9GZeePC+kxGsX6snJv+4hwYtSsT8tOXRe
b0QjZRSAksE5WQqnOOTFK+qpgGcwm1m6hXaVOS/oj4MUUc16cb7mNg8hTokqeBQp
6HUrVysJJxOxkFsJHdaK75WDq5f+XI21bh+VzuGmSrCHjIWtRl03plB2UEaFF90H
mjl2zrf3YpVTJz5dLKdCox4HQ+dmrUfKKzdyyz1z7ubNauWcpghLbcX6wudLKIsv
uxgjjjf9BzG4vL4scMU82qfVzOMul41Tmkl13BQtWCM0Q2HrE7YgIr0vzrXeUwcJ
StAt7+yIoBTyolw4RxU56EM6DUo6lddlWOaGDNahI67LMjHbSFziYSGxUYeduumU
Xq+gS1dypB+nU14qj37cMkLLbNxu1g4d04ghj55h2bGKPGw5ovbnW4/AaHKZmntD
poGStNDiyrcHHT7bszg+uAYIcjidpaqPHnl8qu31TFR7S9FLqyR00RoCZOE0/fls
m3OivE7pARqWW5LJKG630612DvgVcT/Sy/WYKMXRoEsE0iSLE5niNirZZiTamwtz
8NQl6R6AV7GQQMbFlJxl0zlrXBJZ0fQw0Rih8KnBZ7C807Udn4eCYav+L5/+8wb/
v3755731HwG/rv9wlCQw4ob/GPaE/78c+s/jlAA9xgkfSgnQY5zwoZQAPcoJj+o/
T/XwUz38VA8/1cPfl3r4U/6/xC+joG7qPAqqj8r/t9f97vkfRXGKIG/vf5Io8aT/
fhTr8v1d2IQ/fp3+xUOw21fxy6ZN724/BVUexT9+82zwza9Rs7sL5niIRxQZkySF
wHcohkzoDXdnZ6c/mKDIhI/DCTpO9ATB75DFHYFMBM2+u3v+JB99i+P//kt+/JrT
//fqPwhMvH3/m0DJ+/d/Cewp/j+GjTE7+TTQx3wPe/kl0T6xpNUMfQk/h9w8uHvz
bWar1gRZjJ9eoi/xH4ZtUEfZq+siE/BQqYMu23bIeIoWDC/4pu2DdtvdTajRw7cj
gEzoJpz86HH8+cMfHoP0VUBGcyJKnkN2czcBt0Y+nPMoMD2cNrZqjsEIReHPITY4
xneTrwSm5xATVNULib2bfB7Xxh+s+HA3eQ1vzyHpdYjcTT4BRJMhR8vbh7u1fVnL
bBYwuVjY6GCiK9DTbZMTm/lxK08ZFbvmGwrVsc20Ws3jJl9X7XNo8tb85akPYTaP
+IUgDIA49+iOsZ0hgtcCrMjLbD8mz826Pq104G8jn/M6d2kApZhxZrcxF5+9Fp0M
+iq4ejvZL4Qd2/Bs6OjBTjaLnFstlRrBTsCXolOyuGqvPvm0Ry+kOhnHODse993d
bPbuhMxuz4bHAWnq43jGCzWu02N2N4GfMP27gv+7uOuCNP7Iz/9hfPTd8j8SJ0n0
dhxCUOST/vOE/18P/n8otH8vge1z8d9lAfIN3ON98Q9T5Nv6j8TQW/wj5FP+91HM
EgHye5+r/X//1QRPonmyuP0xL6KSKCbRLRXEczJOgnmCJeOO3M4RLIafiP87Fv95
Wsfbjx7/6P3/f1/HPwnf4p8gyaf4/xj2q3/52//yz3/21z//8+Xv/P0PfuOnPxv+
6d9+/fyjX/uVv/nZz6N/+KPf/Y8//ff5X/xP1Ef/ufzf//rkj/1/vfvv3/rH+V/9
5i/+5O9+0Pxt/lNN/ckvDv/XztnGthDHcVwkIo5GgphFws1DkCX2v6fetVF0LXbm
brc+t69cHzbtddfT62oVIvHKCyR4g3hamMQrBJlXEi9MmLAICRLCWxERDxPihWuL
1WwrmdZDfp9XTdepeed+v73/9nat90516uGKjt7HOGromrJ5785EzK
8+07ZrYozOnFB/sOWieZLpkS4da9ddO5M++ePK2t6z/+ngIF/nX6Xx5TI9XVP8N+
0z+LCvM/FINA/9VgnCFGCBtnctGI/UpcATccFdN/XM3GM5WpgMrpn0HUF/0zyFiz
5fXPGpYA+q8CY5c8Y7RDw/ofiqAJxoLzoo/3jNQCUVZEWhFVaIHK5w3WrM1Yfads
Xj2WtuMkZ0EcTTOkufQhL+K8ROOjHcKE6TZ3TNfjKRV3OyUTFrGVfUfGhvKr/g6b
3BmNp3DaQrAk7vJIDXaflH9CtqUzWoesWREuOQRvA2d8ef/xkGC4/itRAZWd/0Xm

```
of1/CLL4/29Y/1eFomDxH4T+VedjJH+j6WjEIJArJnjlMr+v2WBFkr7vPGqkXHK0
pqjEvIoxoRzJWEs/3WiXYuV3hZEjx9P1ql1pSjdRaqZD7mpPI8IjdHk6MgmJFezu
Lcm46Ip2ai6eyGpKwJUNBbqIhGxRkNpI6qUlj0j567OE5NRzFCnIm0NxJDD+tjVK
oCvc7lGloB8l+FQz1yTkgoTPw8baM0hR6EC6ea2jKehf5yg9VlIXLI519KbN9cH1
IY8gu10tKcHbuCmu+PiE2srllJjT7A2ILbk2fozCaITTH9YYeXKacXllTTOumZwx
jLlBj2o/5q4EzZowEwY/AX/C/ytRAZXv/6mh/octzP/QsP8H+P9/4f9gfsA/5P+V
qADLzn8iYmj9T+f9nzEzsP9HVSj0f8Ozn2U2nEEswbahMJItZkJmiDbDSCNsNMpG
OEbOz+nKZDSGSBnU/5/pvwIV4M/3f4b+2cL+PyQL/V9VmLqA+6gtmnf+/bGHh/uw
t/cpQlQiqy5MaJ4SZmrUnhnMlUM1awf91wOPBoUlkxXT49TZe69e9G/fnTgdHrjt
28bXXLs888jg7Lvk0vqrt27s8bb0nujlFz3benNW3YPAxp5zdxrn7kp82Bnc0Lff
+XyaNCe8z7Xw6HRy39OPrbp44zUo8G/U/++tAH++/2OMl+bzfxbB/Hd1GGcmJWLj
DKKS2K+kT3C3AQAAAAAAAAAAAAAAAAAAAAAUI7Ptcw1rAB4AAA=
-- 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-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

Acknowledgment

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