

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: August 18, 2014

J. Peterson
NeuStar
C. Jennings
Cisco
E. Rescorla
RTFM, Inc.
February 14, 2014

Authenticated Identity Management in the Session Initiation Protocol
(SIP)
draft-jennings-stir-rfc4474bis-01.txt

Abstract

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

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <http://datatracker.ietf.org/drafts/current/>.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on August 18, 2014.

Copyright Notice

Copyright (c) 2014 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents

(<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	3
2. Background	3
3. Overview of Operations	5
4. Signature Generation and Validation	6
4.1. Authentication Service Behavior	6
4.1.1. Intermediary Authentication Services	9
4.2. Verifier Behavior	10
4.3. Identity within a Dialog and Retargeting	11
5. Credentials	12
5.1. Credential Use by the Authentication Service	12
5.2. Credential Use by the Verification Service	13
5.3. Handling Identity-Info URIs	14
5.4. Credential Systems	14
6. Identity Types	15
6.1. Telephone Numbers	15
6.2. Usernames with Domain Names	16
7. Header Syntax	17
8. Examples	21
9. Privacy Considerations	21
10. Security Considerations	22
10.1. Handling of digest-string Elements	22
10.2. Securing the Connection to the Authentication Service	24
10.3. Authorization and Transitional Strategies	25
10.4. Display-Names and Identity	26
11. IANA Considerations	27
11.1. Header Field Names	27
11.2. 428 'Use Identity Header' Response Code	27
11.3. 436 'Bad Identity-Info' Response Code	27
11.4. 437 'Unsupported Credential' Response Code	28
11.5. 438 'Invalid Identity Header' Response Code	28
11.6. Identity-Info Parameters	28
11.7. Identity-Info Algorithm Parameter Values	28
12. Acknowledgments	29
13. Changes from RFC4474	29
14. Informative References	29
Authors' Addresses	31

1. Introduction

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

RFC 3261 [1] stipulates several places within a SIP request where a user can express an identity for themselves, notably the user-populated From header field. However, the recipient of a SIP request has no way to verify that the From header field has been populated appropriately, in the absence of some sort of cryptographic authentication mechanism.

RFC 3261 [1] specifies a number of security mechanisms that can be employed by SIP user agents (UAs), including Digest, Transport Layer Security (TLS), and S/MIME (implementations may support other security schemes as well). However, few SIP user agents today support the end-user certificates necessary to authenticate themselves (via S/MIME, for example), and furthermore Digest authentication is limited by the fact that the originator and destination must share a prearranged secret. It is desirable for SIP user agents to be able to send requests to destinations with which they have no previous association -- just as in the telephone network today, one can receive a call from someone with whom one has no previous association, and still have a reasonable assurance that the person's displayed calling party number (and/or Caller-ID) is accurate. A cryptographic approach, like the one described in this document, can provide a much stronger and less spoofable assurance of identity than the telephone network provides today.

2. Background

The usage of many SIP applications and services is governed by authorization policies. These policies may be automated, or they may be applied manually by humans. An example of the latter would be an Internet telephone application that displays the calling party number (and/or Caller-ID) of a caller, which a human may review to make a policy decision before answering a call. An example of the former would be a voicemail service that compares the identity of the caller to a whitelist before determining whether it should allow the caller access to recorded messages. 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 provides a strong identity system for SIP requests in which authorization policies cannot be circumvented by impersonation.

This document proposes an authentication architecture for SIP in which requests are processed by a logical authentication service that may be implemented as part of a user agent or as a proxy server. Once a message has been authenticated, the service then adds new cryptographic information to requests to communicate to other SIP entities that the sending user has been authenticated and its use of the From header field has been authorized.

But authorized by whom? Identities are issued to users by authorities. When a new user becomes associated with example.com, the administrator of the SIP service for that domain will issue them an identity in that namespace, such as alice@example.com. Alice may then send REGISTER requests to example.com that make her user agents eligible to receive requests for sip:alice@example.com. In some cases, Alice may be the owner of the domain herself, and may issue herself identities as she chooses. But ultimately, it is the controller of the SIP service at example.com that must be responsible authorizing the use of names in the example.com domain. Therefore, the credentials needed to prove this authorization must ultimately derive from the domain owner: either a user agent gives requests to the domain name owner in order for them to be signed by the domain owner's credentials, or the user agent must possess credentials that prove in some fashion that the domain owner has given the user agent the right to a name.

The situation is however more complicated for telephone numbers. Authority over telephone numbers does not correspond directly to Internet domains. While a user could register at a SIP domain with a username that corresponds to a telephone number, any connection between the administrator of that domain and the assignment of telephone numbers is not currently reflected on the Internet. Telephone numbers do not share the domain-scope property described above, as they are dialed without any domain component. This document thus assumes the existence of a separate means of establishing authority over telephone numbers, for cases where the telephone number is the identity of the user. As with SIP URIs, the necessary credentials to prove authority for a name might reside either in the endpoint or at some intermediary.

This document specifies a means of sharing a cryptographic assurance of end-user SIP identity in an interdomain or intradomain context that is based on the authentication service adding a SIP header, the Identity header. In order to assist in the validation of this

assurance, this specification also describes an Identity-Info header that can be used by the recipient of a request to recover the credentials of the signer. Note that the scope of this document is limited to providing this identity assurance for SIP requests; solving this problem for SIP responses is outside the scope of this work.

This specification allows either a user agent or a proxy server to provide identity services and to verify identities. To maximize end-to-end security, it is obviously preferable for end-users to acquire their own credentials; if they do, their user agents can act as an authentication service. However, end-user credentials may be neither practical nor affordable, given the potentially large number of SIP user agents (phones, PCs, laptops, PDAs, gaming devices) that may be employed by a single user. In such environments, synchronizing keying material across multiple devices may be very complex and requires quite a good deal of additional endpoint behavior. Managing several credentials for the various devices could also be burdensome. This trade-off needs to be understood by implementers of this specification.

3. Overview of Operations

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

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

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

The authentication service authenticates Alice (possibly by sending a Digest authentication challenge) and validates that she is authorized to assert the identity that is populated in the From header field. This value may be Alice's AoR, or in other cases it may be some different value that the proxy server has authority over, such as a telephone number. It then computes a hash over some particular headers, including the From header field (and optionally the body) of the message. This hash is signed with the appropriate credential (example.com, in the sip:alice@example.com case) and inserted in a new header field in the SIP message, the 'Identity' header.

The proxy, as the holder of the private key for its domain, is asserting that the originator of this request has been authenticated and that she is authorized to claim the identity (the SIP address-

of-record) that appears in the From header field. The proxy also inserts a companion header field, Identity-Info, that tells Bob how to acquire keying material necessary to validate its credentials, if he doesn't already have it.

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

4. Signature Generation and Validation

4.1. Authentication Service Behavior

This document specifies a role for SIP entities called an authentication service. The authentication service role can be instantiated by an intermediary such as a proxy server or a user agent. Any entity that instantiates the authentication service role MUST possess the private key of one or more credentials that can be used to sign for a domain or a telephone number (see Section 5.1). Intermediaries that instantiate this role MUST be capable of authenticating one or more SIP users who can register for that identity. Commonly, this role will be instantiated by a proxy server, since these entities are more likely to have a static hostname, hold corresponding credentials, and have access to SIP registrar capabilities that allow them to authenticate users. It is also possible that the authentication service role might be instantiated by an entity that acts as a redirect server, but that is left as a topic for future work.

SIP entities that act as an authentication service MUST add a Date header field to SIP requests if one is not already present (see Section 7 for information on how the Date header field assists verifiers).

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

Step 1:

The authentication service MUST extract the identity of the sender from the request. The authentication service takes this value from the From header field; this AoR will be referred to here as the 'identity field'. If the identity field contains a SIP or SIP Secure (SIPS) URI, and the user portion is not a telephone number, the

authentication service MUST extract the hostname portion of the identity field and compare it to the domain(s) for which it is responsible (following the procedures in RFC 3261 [1], Section 16.4), used by a proxy server to determine the domain(s) for which it is responsible). If the identity field uses the TEL URI scheme, or the identity field is a SIP or SIPS URI with a telephone number in the user portion, the authentication service determines whether or not it is responsible for this telephone number; see Section 6.1 for more information. If the authentication service is not authoritative for the identity in question, it SHOULD process and forward the request normally, but it MUST NOT following the steps below to add an Identity header; see below for more information on authentication service handling of an existing Identity header. [where?]

Step 2:

The authentication service MUST then 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 [1], 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 credential, or preferably by providing a password that unlocks said private key.

Authorization of the use of a particular username or telephone number in the user part of the From header field is a matter of local policy for the authentication service, see Section 5.1 for more information.

Note that this check is performed only on the addr-spec in the From header field (e.g., the URI of the sender, like 'sip:alice@atlanta.example.com'); it does not convert the display-name portion of the From header field (e.g., 'Alice Atlanta').

Authentication services MAY check and validate the display-name as well, and compare it to a list of acceptable display-names that may be used by the sender; if the display-name does not meet policy constraints, the authentication service MUST return a 403 response code. The reason phrase should indicate the nature of the problem; for example, "Inappropriate Display Name". However, the display-name is not always present, and in many environments the requisite operational procedures for display-name validation may not exist. For more information, see Section 10.4.

Step 3:

The authentication service SHOULD ensure that any preexisting Date header in the request is accurate. Local policy can dictate precisely how accurate the Date must be; a RECOMMENDED maximum discrepancy of ten minutes will ensure that the request is unlikely to upset any verifiers. If the Date header contains a time different by more than ten minutes from the current time noted by the authentication service, the authentication service SHOULD reject the request. This behavior is not mandatory because a user agent client (UAC) could only exploit the Date header in order to cause a request to fail verification; the Identity header is not intended to provide a source of non-repudiation or a perfect record of when messages are processed. Finally, the authentication service MUST verify that the Date header falls within the validity period of its credential. For more information on the security properties associated with the Date header field value, see Section 7.

[TBD: Should consider a lower threshold than ten minutes? With the removal of other elements from the sig, that's a lot of leeway.]

Step 4:

The authentication service MAY form an identity-reliance signature and add an Identity-Reliance header to the request containing this signature. The Identity-Reliance header provides body security properties that are useful for non-INVITE transactions, and in environments where body security of INVITE transactions is necessary. Details on the generation of this header is provided in Section 7. If the authentication service is adding an Identity-Reliance header, it MUST also add a Content-Length header field to SIP requests if one is not already present; this can help verifiers to double-check that they are hashing exactly as many bytes of message-body as the authentication service when they verify the message.

Step 5:

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 credential can be acquired; see Section 5.3 for more on credential acquisition. Details on the syntax of both of these headers are provided in Section 7.

Finally, the authentication service MUST forward the message normally.

4.1.1.1. Intermediary Authentication Services

In cases where a user agent does not possess its own credentials to sign an Identity header, the user agent can send its request through an intermediary that will provide a signed Identity header based on the contents of the request. This requires, among other things, that intermediaries have some means of authenticating the user agents sending requests.

All RFC 3261 [1] compliant user agents support Digest authentication, which utilizes a shared secret, as a means for authenticating themselves to a SIP registrar. Registration allows a user agent to express that it is an appropriate entity to which requests should be sent for a particular SIP AoR URI (e.g., 'sip:alice@atlanta.example.com'). For such SIP URIs, by the definition of identity used in this document, registration proves the identity of the user to a registrar. Similar checks might be performed for telephone numbers as identities. 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.

RFC 3261 [1] already describes an intermediary architecture very similar to the one proposed in this document 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.

This specification assumes that UACs will have an appropriate means to discover an authentication service that can sign with a credential

corresponding to the UAC's identity. Most likely, this information will simply be provisioned in UACs.

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 RFC 3325 [9]. 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.

4.2. Verifier Behavior

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

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

Step 1:

In order to determine whether the signature for the URI in the From header field value should be over the entire URI or just a canonicalized telephone number, the verification service must follow the process described in Section 6.1. That section also describes the procedures the verification service must follow to determine if the signer is authoritative for a telephone number. For domains, the verifier MUST follow the process described in Section 6.2 to determine if the signer is authoritative for the URI in the From header field.

Step 2:

The verifier must first ensure that it possesses the proper keying material to validate the signature in the Identity header field. See Section 5.2 for more information on these procedures.

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 7. If a verifier determines that the signature on the message does not correspond to the reconstructed digest-string, then a 438 'Invalid Identity Header' response MUST be returned.

Step 4:

If the request contains an Identity-Reliance header, the verifier SHOULD verify the signature in the Identity-Reliance header field, following the procedures for generating the hashed reliance-digest-string described in Section 7. If a verifier determines that the signature on the message does not correspond to the reconstructed digest-string, then a 438 'Invalid Identity Header' response SHOULD be returned.

Step 5:

The verifier MUST validate the Date header in the manner described in Section 10.1; recipients that wish to verify Identity signatures MUST support all of the operations described there. It must furthermore ensure that the value of the Date header falls within the validity period of the credential used to sign the Identity header.

4.3. Identity within a Dialog and Retargeting

The mechanism in this document provides a signature over the URI in the To header field value. The recipient of a request must compare that value to their own identity in order to determine whether or not the identity information in this call might have been replayed. Retargeting, however, complicates this evaluation.

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

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. Moreover, 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. But in retargeting cases, if the URI in

the From header does not identify the sender of the request in the backwards direction, then clearly it would be inappropriate to provide an Identity signature over that From header. As specified above, if the authentication service is not responsible for the domain in the From header field of the request, it MUST NOT add an Identity header to the request, and it should process/forward the request normally.

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

For further information on the problems of response identity see [17].

5. Credentials

5.1. Credential Use by the Authentication Service

In order to act as an authentication service, a SIP entity must have access to the private keying material of one or more credentials that cover URIs, domain names or telephone numbers. These credentials may represent authority over only a single name (such as `alice@example.com`), an entire domain (such as `example.com`), or potentially a set of domains. Similarly, a credential may represent authority over a single telephone number or a range of telephone numbers. The way that the scope of a credential is expressed is specific to the credential mechanism.

Authorization of the use of a particular username or telephone number in the user part of the From header field is a matter of local policy for the authentication service, one that depends greatly on the manner in which authentication is performed. For non-telephone number user parts, one policy might be as follows: the username given in the 'username' parameter of the Proxy-Authorization header MUST correspond exactly to the username in the From header field of the SIP message. However, there are many cases in which this is too limiting or inappropriate; a realm might use 'username' parameters in Proxy-Authorization that do not correspond to the user-portion of SIP From headers, or a user might manage multiple accounts in the same

administrative domain. In this latter case, a domain might maintain a mapping between the values in the 'username' parameter of Proxy-Authorization and a set of one or more SIP URIs that might legitimately be asserted for that 'username'. For example, the username can correspond to the 'private identity' as defined in Third Generation Partnership Project (3GPP), in which case the From header field can contain any one of the public identities associated with this private identity. In this instance, another policy might be as follows: the URI in the From header field MUST correspond exactly to one of the mapped URIs associated with the 'username' given in the Proxy-Authorization header. This is a suitable approach for telephone numbers in particular. 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.

5.2. Credential Use by the Verification Service

In order to act as a verification service, a SIP entity must have a way to acquire and retain credentials for authorities over particular URIs, domain names and/or telephone numbers. The Identity-Info header (as described in the next section) is supported by all verification service implementations to create a baseline means of credential acquisition. Provided that the credential used to sign a message is not previously known to the verifier, SIP entities SHOULD discover this credential by dereferencing the Identity-Info header, unless they have some more efficient implementation-specific way of acquiring certificates. If the URI scheme in the Identity-Info header cannot be dereferenced, then a 436 'Bad Identity-Info' response MUST be returned.

Verification service implementations supporting this specification SHOULD have some means of retaining credentials (in accordance with normal practices for credential lifetimes and revocation) in order to prevent themselves from needlessly downloading the same credential every time a request from the same identity is received. Credentials cached in this manner may be indexed in accordance with local policy: for example, by their scope, or the URI given in the Identity-Info header field value.

[TBD: Should we add some kind of hash or similar indication to the Identity-Info header to make it easier for verifiers to ascertain that they already possess a credential without dereferencing the URI?]

5.3. Handling Identity-Info URIs

An Identity-Info header MUST contain a URI which dereferences to a resource which contains the public key components of the credential used by the authentication service to sign a request. Much as is the case with the trust anchor(s) required for deployments of this specification, it is essential that a URI in the Identity-Info header be dereferencable by any entity that could plausibly receive the request. For common cases, this means that the URI must be dereferencable by any entity on the public Internet. In constrained deployment environments, a service private to the environment might be used instead.

Beyond providing a means of accessing credentials for an identity, the Identity-Info header further services a means of differentiating which particular credential was used to sign a request, when there are potentially multiple authorities eligible to sign. For example, imagine a case where a domain implements the authentication service role for example.com, and a user agent belonging to Alice has acquired a credential for alice@example.com. Either would be eligible to sign a SIP request from alice@example.com. Verification services however need a means to differentiate which one performed the signature. The Identity-Info header performs that function.

5.4. Credential Systems

This document makes no specific recommendation for the use of any credential system. Today, there are two primary credential systems in place for proving ownership of domain names: certificates (e.g., X.509 v3, see [8]) and the domain name system itself (e.g., DANE, see [10]). It is envisioned that either could be used in the SIP context: an Identity-Info header could for example give an HTTP URL of the form 'application/pkix-cert' pointing to a certificate (following the conventions of [3]). The Identity-Info headers may use the DNS URL scheme (see [11]) to indicate keys in the DNS.

While no comparable public credentials exist for telephone numbers, either approach could be applied to telephone numbers. A credential system based on certificates is given in draft-peterson-stir-certificates [TBD - fix after submitting]. One based on the domain name system is given in [18].

In order for a credential system to work with this mechanism, its specification must detail:

which URIs schemes the credential will use in the Identity-Info header, and any special procedures required to dereference the URIs

how the verifier can learn the scope of the credential.

any special procedures required to extract keying material from the resources designated by the URI

any algorithms that would appear in the Identity-Info "alg" parameter other than 'rsa-sha256.' Note that per the IANA Considerations of this document (Section 11.7), new algorithms can only be specified by Standards Action.

SIP entities cannot reliably predict where SIP requests will terminate. When choosing a credential scheme for deployments of this specification, it is therefore essential that the trust anchor(s) for credentials be widely trusted, or that deployments restrict the use of this mechanism to environments where the reliance on particular trust anchors is assured by business arrangements or similar constraints.

Note that credential systems must address key lifecycle management concerns: were a domain to change the credential 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 credential, 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).

[TBD: What will the normative language here be? Support for which mechanisms?]

6. Identity Types

6.1. Telephone Numbers

Since many SIP applications provide a Voice over IP (VoIP) service, telephone numbers are commonly used as identities in SIP deployments. In order for telephone numbers to be used with the mechanism described in this document, authentication services must enroll with an authority that issues credentials for telephone numbers or telephone number ranges, and verification services must trust the authority employed by the authentication service that signs a request. Enrollment procedures and credential management are outside the scope of this document.

Given the existence of such authorities, authentication and verification services must identify when a request should be signed by an authority for a telephone number, and when it should be signed

by an authority for a domain. Telephone numbers most commonly appear in SIP header field values in the username portion of a SIP URI (e.g., 'sip:+17005551008@chicago.example.com;user=phone'). The user part of that URI conforms to the syntax of the TEL URI scheme (RFC 3966 [5]). 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 both of these cases, it's clear that the signer must have authority over the telephone number, not the domain name of the SIP URI. It is also possible, however, for requests to contain a URI like 'sip:7005551000@chicago.example.com'. It may be non-trivial for a service to ascertain in this case whether the URI contains a telephone number or not.

To address this problem, the authentication service and verification service both must perform the following canonicalization procedure on any SIP URI they inspect which contains a wholly numeric user part.

[TBD canonicalization algorithm - drop the characters, +'s, assess if its a valid local number (if so, append country code), etc]

[TBD define tn-spec here for ABNF purposes]

If the result of this procedure forms a complete telephone number, that number is used for the purpose of creating and signing the digest-string by both the authentication service and verification service. If the result does not form a complete telephone number, the authentication service and verification service should treat the entire URI as a SIP URI, and apply a domain signature per the procedures in Section 6.2.

In the longer term, it is possible that some directory or other discovery mechanism 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.

6.2. Usernames with Domain Names

When a verifier processes a request containing an Identity-Info header with a domain signature, it must compare the domain portion of the URI in the From header field of the request with the domain name that is the subject of the credential acquired from the Identity-Info header. While this might seem that this should be a straightforward process, it is complicated by two deployment realities. In the first place, credentials have varying ways of describing their subjects, and may indeed have multiple subjects, especially in 'virtual hosting' cases where multiple domains are managed by a single application. Secondly, some SIP services may delegate SIP functions

to a subordinate domain and utilize the procedures in RFC 3263 [2] that allow requests for, say, 'example.com' to be routed to 'sip.example.com'. As a result, a user with the AoR 'sip:jon@example.com' may process requests through a host like 'sip.example.com', and it may be that latter host that acts as an authentication service.

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

A verifier **MUST** evaluate the correspondence between the user's identity and the signing credential by following the procedures defined in RFC 2818 [7], Section 3.1. While RFC 2818 [7] deals with the use of HTTP in TLS and is specific to certificates, the procedures described are applicable to verifying identity if one substitutes the "hostname of the server" in HTTP for the domain portion of the user's identity in the From header field of a SIP request with an Identity header.

7. Header Syntax

This document specifies three SIP headers: Identity, Identity-Reliance and Identity-Info. Each of these headers can appear only once in a SIP request; Identity-Reliance is **OPTIONAL**, while Identity and Identity-Info are **REQUIRED** for securing requests with this specification. The grammar for these three headers is (following the ABNF [12] in RFC 3261 [1]):

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

```
Identity-Reliance = "Identity-Reliance" HCOLON signed-identity-reliance-diges
t signed-identity-reliance-digest = LDQUOT 32LHEX RDQUOT
```

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

[TBD: The version has the Identity-Reliance header covered under the Identity signature. It is also possible to do this the other way around, where the base Identity signature is generated first, and Identity-Reliance would cover both the Identity header and the body. This is a trade-off of whether the authentication service should decide whether Identity-Reliance is needed or if the verification service should decide. These have different properties, and some investigation would be needed to decide between them.]

The signed-identity-reliance-digest is a signed hash of a canonical string generated from certain components of a SIP request. Creating this hash and the Identity-Reliance header field to contain it is OPTIONAL, and its usage is a matter of local policy for authentication services. To create the contents of the signed-identity-reliance-digest, the following element of a SIP message MUST be placed in a bit-exact 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.

[TBD: Explore alternatives to including the whole body for INVITE requests; should there be a special case for security parameters that would appear in SDP?]

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

First, the identity. If the user part of the AoR in the From header field of the request contains a telephone number, then the canonicalization of that number goes into the first slot (see Section 6.1). Otherwise, the first slot contains the AoR of the UA sending the message, or addr-spec of the From header field.

Second, the target. If the user part of the AoR in the To header field of the request contains a telephone number, then the canonicalization of that number goes into the second slot (see Section 6.1). Otherwise, the second slot contains the addr-spec component of the To header field, which is the AoR to which the request is being sent.

Third, the request method.

Fourth, the Date header field, with exactly one space each for each SP and the weekday and month items case set as shown in the BNF of RFC 3261 [1]. RFC 3261 specifies that the BNF for weekday and month is a choice amongst a set of tokens. The RFC 4234 [12] rules for the BNF specify that tokens are case sensitive. However, when used to construct the canonical string defined here, the first letter of each week and month MUST be capitalized, and the remaining two letters must be lowercase. This matches the capitalization provided in the definition of each token. All requests that use the Identity mechanism MUST contain a Date header.

Fifth, the Identity-Reliance header field value, if there is an Identity-Reliance field in the request. If the message has no body, or no Identity-Reliance header, then the fifth slot 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 10 and [4]. The precise formulation of this digest-string is, therefore (following the ABNF[12] in RFC 3261 [1]):

```
digest-string = addr-spec / tn-spec "|" addr-spec / tn-spec "|"
                Method "|" SIP-date "|" [ signed-identity-reliance-digest ]
```

For the definition of 'tn-spec' see Section 6.1.

After the digest-string or reliance-digest-string is formed, each MUST be hashed and signed with the certificate of authority over the identity. 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-sha256'; further values MUST be defined in a Standards Track RFC, see Section 14.7 for more information. All implementations of this specification MUST support 'rsa-sha256'. When the 'rsa-sha256' 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 [13] and base64 encode the results as specified in RFC 3548 [14]. A 2048-bit or longer RSA key MUST be used. The result of the digest-string hash is placed in the Identity header field; the optional reliance-digest-string hash goes in the Identity-Reliance header. For detailed examples of the usage of this algorithm, see Section 8.

The 'absoluteURI' portion of the Identity-Info header MUST contain a URI; see Section 5.3 for more on choosing how to advertise credentials through Identity-Info.

This document adds (or amends) the following entries to Table 2 of RFC 3261 [1] (this repeats the registrations of RFC4474):

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
Header field	where	proxy	ACK	BYE	CAN	INV	OPT	REG
-----	----	-----	----	----	----	----	----	----
Identity-Reliance	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. The Identity and Identity-Info header MUST NOT appear in CANCEL. Note as well that the use of Identity with REGISTER is consequently a subject for future study, although it is left as optional here for forward-compatibility reasons.

8. Examples

9. Privacy Considerations

The identity mechanism presented in this document is compatible with the standard SIP practices for privacy described in RFC 3323 [15]. A SIP proxy server can act both as a privacy service and as an authentication service. Since a user agent can provide any From header field value that the authentication service is willing to authorize, there is no reason why private SIP URIs that contain legitimate domains (e.g., sip:anonymous@example.com) cannot be signed by an authentication service. The construction of the Identity header is the same for private URIs as it is for any other sort of URIs.

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

RFC 3325 [9] defines the "id" priv-value token, which is specific to the P-Asserted-Identity header. The sort of assertion provided by the P-Asserted-Identity header is very different from the Identity header presented in this document. It contains additional information about the sender of a message that may go beyond what appears in the From header field; P-Asserted-Identity holds a definitive identity for the sender that is somehow known to a closed network of intermediaries that presumably the network will use this identity for billing or security purposes. The danger of this network-specific information leaking outside of the closed network motivated the "id" priv-value token. The "id" priv-value token has no implications for the Identity header, and privacy services MUST NOT remove the Identity header when a priv-value of "id" appears in a Privacy header.

Finally, note that unlike RFC 3325 [9], the mechanism described in this specification adds no information to SIP requests that has privacy implications.

10. Security Considerations

10.1. Handling of digest-string Elements

This document describes a mechanism that provides a signature over the Date header field, and either the whole or part of the To and From header fields of SIP requests, as well as optional protections for the message body. 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 replayed in cut-and-paste attacks. In general, the considerations related to the security of these headers are the same as those given in RFC 3261 [1] for including headers in tunneled 'message/sip' MIME bodies (see Section 23 in particular). The following section details the individual security properties obtained by including each of these header fields within the signature; collectively, this set of header fields provides the necessary properties to prevent impersonation.

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

The Date header field provides replay protection, as described in RFC 3261 [1], 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). The result of this is that if an Identity header is replayed within the Date interval, verifiers will recognize that it is invalid; if an Identity header is replayed after the Date interval, verifiers will recognize that it is invalid because the Date is stale.

Without the method, an INVITE request could be cut- and-pasted by an attacker and transformed into a MESSAGE 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.

RFC4474 originally had protections for the Contact, Call-ID and CSeq. These are removed from RFC4474bis. The absence of these header values creates some opportunities for determined attackers to impersonate based on cut-and-paste attacks; however, the absence of these headers does not seem impactful to preventing against the simple unauthorized claiming of a From header field value, which is the primary scope of the current document.

It might seem attractive to provide a signature over some of the information present in the Via header field value(s). For example, without a signature over the sent-by field of the topmost Via header, an attacker could remove that Via header and insert its own in a cut-and-paste attack, which would cause all responses to the request to be routed to a host of the attacker's choosing. However, a signature over the topmost Via header does not prevent attacks of this nature, since the attacker could leave the topmost Via intact and merely insert a new Via header field directly after it, which would cause responses to be routed to the attacker's host "on their way" to the valid host, which has exactly the same end result. Although it is possible that an intermediary-based authentication service could guarantee that no Via hops are inserted between the sending user agent and the authentication service, it could not prevent an attacker from adding a Via hop after the authentication service, and thereby preempting responses. It is necessary for the proper operation of SIP for subsequent intermediaries to be capable of inserting such Via header fields, and thus it cannot be prevented. As such, though it is desirable, securing Via is not possible through the sort of identity mechanism described in this document; the best known practice for securing Via is the use of SIPS.

This mechanism also provides an optional signature over the bodies of SIP requests. This can help to protect non-INVITE transactions such as MESSAGE or NOTIFY, as well as INVITEs in those environments where intermediaries do not change SDP. While this is not strictly necessary to prevent the impersonation attacks, there is little purpose in establishing the identity of the user that originated a SIP request if this assurance is not coupled with a comparable assurance over the contents of the message. There are furthermore some baiting attacks (where the attacker receives a request from the target and reoriginates it to a third party) that might not be prevented by only a signature over the From, To and Date, but could be prevented by securing SDP. Note, however, that this is not perfect end-to-end security. The authentication service itself, when instantiated at an intermediary, could conceivably change the body (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 bodies, it does not eliminate it entirely. Since it is a foundational assumption of this mechanism that the users trust their local domain to vouch for their security, they must also trust the service not to violate the integrity of their message without good reason.

In the end analysis, the Identity, Identity-Reliance 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 the originators of SIP requests can prove that they are who they claim to be. At best, by stripping identity information from a request, a man-in-the-middle could make it impossible to distinguish any illegitimate messages he would like to send from those messages sent by an authorized user. However, it requires a considerably greater amount of energy to mount such an attack than it does to mount trivial impersonations by just copying someone else's From header field. This mechanism provides a way that an authorized user can provide a definitive assurance of his identity that an unauthorized user, an impersonator, cannot.

One additional respect in which the Identity-Info header cannot protect itself is the 'alg' parameter. The 'alg' parameter is not included in the digest-string, and accordingly, a man-in-the-middle might attempt to modify the 'alg' parameter. Once again, it is important to note that preventing men-in-the-middle is not the primary impetus for this mechanism. Moreover, changing the 'alg' would at worst result in some sort of bid-down attack, and at best cause a failure in the verifier. Note that only one valid 'alg' parameter is defined in this document and that thus there is currently no weaker algorithm to which the mechanism can be bid down. 'alg' has been incorporated into this mechanism for forward-compatibility reasons in case the current algorithm exhibits weaknesses, and requires swift replacement, in the future.

10.2. Securing the Connection to the Authentication Service

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

If a user does not receive a certificate from the authentication service over this TLS connection that corresponds to the expected domain (especially when the user receives a challenge via a

mechanism such as Digest), then it is possible that a rogue server is attempting to pose as an authentication service for a domain that it does not control, possibly in an attempt to collect shared secrets for that domain. A similar practice could be used for telephone numbers, though the application of certificates for telephone numbers to TLS is left as a matter for future study.

Without TLS, the various header field values and the body of the request will not have integrity protection when the request arrives at an authentication service. Accordingly, a prior legitimate or illegitimate intermediary could modify the message arbitrarily.

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

Accordingly, direct TLS connections SHOULD be used between the UAC and the authentication service whenever possible. The opportunistic nature of this mechanism, however, makes it very difficult to constrain UAC behavior, and moreover there will be some deployment architectures where a direct connection is simply infeasible and the UAC cannot act as an authentication service itself. Accordingly, when a direct connection and TLS are not possible, a UAC should use the SIPs mechanism, Digest 'auth-int' for body integrity, or both when it can. The ultimate decision to add an Identity header to a request lies with the authentication service, of course; domain policy must identify those cases where the UAC's security association with the authentication service is too weak.

10.3. Authorization and Transitional Strategies

Ultimately, the worth of an assurance provided by an Identity header is limited by the security practices of the authentication service that issues the assurance. Relying on an Identity header generated by a remote administrative domain assumes that the issuing domain uses recommended administrative practices to authenticate its users. However, it is possible that some authentication services 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 authentication services is questionable. While there is no magic way for a verifier to distinguish "good" from "bad" signers 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.

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 or telephone number that uses Identity, and in the future, view messages received from that sources without Identity headers with skepticism.

A verifier could consult some sort of directory that indications whether a given caller should have a signed identity. There are a number of potential ways in which this could be implemented. This is left as a subject for future work.

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

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

10.4. 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. Securing the display-name component of the From header field value is outside the scope of this document, but may be the subject of future work.

11. IANA Considerations

[TBD: update for rfc4474bis or remove?]

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.

11.1. Header Field Names

This document specifies three SIP headers: Identity, Identity-Reliance and Identity- Info. Their syntax is given in Section 7. These headers are defined by the following information, which has been added to the header sub-registry under <http://www.iana.org/assignments/sip-parameters>

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

11.2. 428 'Use Identity Header' Response Code

This document registers a SIP response code, which is described in Section 4.2. It is sent when a verifier receives a SIP request that lacks an Identity header in order to indicate that the request should be re-sent with an Identity header. This response code is defined by the following information, which has been added to the method and response-code sub-registry under <http://www.iana.org/assignments/sip-parameters>

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

11.3. 436 'Bad Identity-Info' Response Code

This document registers a SIP response code, which is described in Section 4.2. It is used when the Identity-Info header contains a URI that cannot be dereferenced by the verifier (either the URI scheme is unsupported by the verifier, or the resource designated by the URI is otherwise unavailable). This response code is defined by the following information, which has been added to the method and response-code sub-registry under <http://www.iana.org/assignments/sip-parameters>

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

11.4. 437 'Unsupported Credential' Response Code

This document registers a SIP response code, which is described in Section 4.2. It is used when the verifier cannot validate the credential referenced by the URI of the Identity-Info header, because, for example, the credential is self-signed, or signed by an authority for whom the verifier does not trust. This response code is defined by the following information, which has been added to the method and response-code sub-registry under <http://www.iana.org/assignments/sip-parameters>

Response Code Number: 437
Default Reason Phrase: Unsupported Credential

11.5. 438 'Invalid Identity Header' Response Code

This document registers a SIP response code, which is described in Section 4.2. It is used when the verifier receives a message with an Identity signature that does not correspond to the digest-string calculated by the verifier. This response code is defined by the following information, which has been added to the method and response-code sub-registry under <http://www.iana.org/assignments/sip-parameters>

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

11.6. Identity-Info Parameters

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

11.7. Identity-Info Algorithm Parameter Values

The IANA has created a registry for Identity-Info 'alg' parameter values. This registry is to be prepopulated with a single entry for a value called 'rsa-sha256', which describes the algorithm used to create the signature that appears in the Identity header. Registry entries must contain the name of the 'alg' parameter value and the

specification in which the value is described. New values for the 'alg' parameter may be defined only in Standards Track RFCs.

A previous version of this specification defined the 'rsa-sha1' value for this registry. That value is hereby deprecated, and should be removed. It is not believed that any implementations are making use of this value.

[TBD - consider EC for smaller credential sizes?]

12. Acknowledgments

Lots of people made significant contributions to this document.

13. Changes from RFC4474

Lots of people made significant contributions to this document.

Generalized the credential mechanism; credential enrollment and acquisition is now outside the scope of this document

Reduced the scope of the Identity signature to remove CSeq, Call-ID, Contact, and the message body.

Added the Identity-Reliance header

Deprecated 'rsa-sha1' in favor of new baseline signing algorithm

[TBD - more]

14. Informative 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] Rosenberg, J. and H. Schulzrinne, "Session Initiation Protocol (SIP): Locating SIP Servers", RFC 3263, June 2002.
- [3] Housley, R. and P. Hoffman, "Internet X.509 Public Key Infrastructure Operational Protocols: FTP and HTTP", RFC 2585, May 1999.
- [4] Peterson, J., "Session Initiation Protocol (SIP) Authenticated Identity Body (AIB) Format", RFC 3893, September 2004.

- [5] Schulzrinne, H., "The tel URI for Telephone Numbers", RFC 3966, December 2004.
- [6] Housley, R., Polk, W., Ford, W., and D. Solo, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 3280, April 2002.
- [7] Rescorla, E., "HTTP Over TLS", RFC 2818, May 2000.
- [8] Cooper, D., Santesson, S., Farrell, S., Boeyen, S., Housley, R., and W. Polk, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 5280, May 2008.
- [9] 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.
- [10] Hoffman, P. and J. Schlyter, "The DNS-Based Authentication of Named Entities (DANE) Transport Layer Security (TLS) Protocol: TLSA", RFC 6698, August 2012.
- [11] Josefsson, S., "Domain Name System Uniform Resource Identifiers", RFC 4501, May 2006.
- [12] Crocker, D., Ed. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", RFC 4234, October 2005.
- [13] Housley, R., "Cryptographic Message Syntax (CMS) Algorithms", RFC 3370, August 2002.
- [14] Josefsson, S., "The Base16, Base32, and Base64 Data Encodings", RFC 3548, July 2003.
- [15] Peterson, J., "A Privacy Mechanism for the Session Initiation Protocol (SIP)", RFC 3323, November 2002.
- [16] Peterson, J., Schulzrinne, H., and H. Tschofenig, "Secure Telephone Identity Problem Statement", draft-ietf-stir-problem-statement-03 (work in progress), January 2014.
- [17] Peterson, J., "Retargeting and Security in SIP: A Framework and Requirements", draft-peterson-sipping-retarget-00 (work in progress), February 2005.

- [18] Kaplan, H., "A proposal for Caller Identity in a DNS-based Entrusted Registry (CIDER)", draft-kaplan-stir-cider-00 (work in progress), July 2013.

Authors' Addresses

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

Email: jon.peterson@neustar.biz

Cullen Jennings
Cisco
400 3rd Avenue SW, Suite 350
Calgary, AB T2P 4H2
Canada

Email: fluffy@iii.ca

Eric Rescorla
RTFM, Inc.
2064 Edgewood Drive
Palo Alto, CA 94303
USA

Phone: +1 650 678 2350
Email: ekr@rtfm.com

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: August 18, 2014

J. Peterson
NeuStar
S. Turner
IECA
February 14, 2014

Secure Telephone Identity Credentials: Certificates
draft-peterson-stir-certificates-00.txt

Abstract

In order to provide a means of proving ownership of telephone numbers on the Internet, some kind of public structure needs to exist that binds cryptographic keys to authority over telephone numbers. This document describes a certificate-based credential system for telephone numbers, which could be used as a part of a broader architecture for managing telephone numbers as identities in protocols like SIP.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <http://datatracker.ietf.org/drafts/current/>.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on August 18, 2014.

Copyright Notice

Copyright (c) 2014 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must

include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	2
2. Terminology	3
3. Enrollment and Authorization	3
3.1. Certificate Scope and Structure	4
3.2. Provisioning Private Keying Material	5
4. Acquiring Credentials to Verify Signatures	5
4.1. Verifying Certificate Scope	6
4.2. Certificate Freshness and Revocation	8
5. Acknowledgments	8
6. IANA Considerations	8
7. Security Considerations	8
8. Informative References	8
Authors' Addresses	10

1. Introduction

As is discussed in the STIR problem statement [13], the primary enabler of robocalling, vishing, swatting and related attacks is the capability to impersonate a calling party number. The starkest examples of these attacks are cases where automated callees on the PSTN rely on the calling number as a security measure, for example to access a voicemail system. Robocallers use impersonation as a means of obscuring identity; while robocallers can, in the ordinary PSTN, block (that is, withhold) their caller identity, callees are less likely to pick up calls from blocked identities, and therefore appearing to calling from some number, any number, is preferable. Robocallers however prefer not to call from a number that can trace back to the robocaller, and therefore they impersonate numbers that are not assigned to them.

One of the most important components of a system to prevent impersonation is an authority responsible for issuing credentials to parties who control telephone numbers. With these credentials, parties can prove that they are in fact authorized to use telephony numbers, and thus distinguish themselves from impersonators unable to present credentials. This document describes a credential system for telephone numbers based on X.509 version 3 certificates in accordance with [7]. While telephone numbers have long been a part of the X.509 standard, the certificates described in this document may contain telephone number blocks or ranges, and accordingly it uses an alternate syntax.

In the STIR in-band architecture, two basic types of entities need access to these credentials: authentication services, and verification services (or verifiers); see [15]. An authentication service must be operated by an entity enrolled with the certificate authority (see Section 3), whereas a verifier need only trust the root certificate of the authority, and have a means to acquire and validate certificates.

The STIR out-of-band architecture is not considered in this document. [TBD]

2. Terminology

In this document, the key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" are to be interpreted as described in RFC 2119 [1] and RFC 6919 [2].

3. Enrollment and Authorization

This document assumes a threefold model for certificate enrollment.

The first enrollment model is one where the certificate authority acts in concert with national numbering authorities to issue credentials to those parties to whom numbers are assigned. In the United States, for example, telephone number blocks are assigned to Local Exchange Carriers (LECs) by the North American Numbering Plan Administrator (NANPA), who is in turn directed by the national regulator. LECs may also receive numbers in smaller allocations, through number pooling, or via an individual assignment through number portability. LECs assign numbers to customers, who may be private individuals or organizations - and organizations take responsibility for assigning numbers within their own enterprise.

The second enrollment model is one where a certificate authority requires that an entity prove control by means of some sort of test. For example, an authority might send a text message to a telephone number containing a URL (which might be deferred by the recipient) as a means of verifying that a user has control of terminal corresponding to that number. Checks of this form are frequently used in commercial systems today to validate telephone numbers provided by users. This is comparable to existing enrollment systems used by some certificate authorities for issuing S/MIME credentials for email by verifying that the party applying for a credential receives mail at the email address in question.

The third enrollment model is delegation: that is, the holder of a certificate (assigned by either of the two methods above) may

delegate some or all of their authority to another party. In some cases, multiple levels of delegation could occur: a LEC, for example, might delegate authority to customer organization for a block of 100 numbers, and the organization might in turn delegate authority for a particular number to an individual employee. This is analogous to delegation of organizational identities in traditional hierarchical PKIs who use the name constraints extension [3]; the root CA delegates names in sales to the sales department CA, names in development to the development CA, etc. As lengthy certificate delegation chains are brittle, however, and can cause delays in the verification process, this document considers optimizations to reduce the complexity of verification.

[TBD] Future versions of this specification will also discuss methods of partial delegation, where certificate holders delegate only part of their authority. For example, an individual assignee may want to delegate authority to an entity for text messages associated with their telephone number, but not for other functions.

3.1. Certificate Scope and Structure

The subjects of telephone number certificates are the administrative entities to whom numbers are assigned or delegated. For example, a LEC might hold a certificate for a range of telephone numbers.

This specification places no limits on the number of telephone numbers that can be associated with any given certificate. Some service providers may be assigned millions of numbers, and may wish to have a single certificate that is capable of signing for any one of those numbers. Others may wish to compartmentalize authority over subsets of the numbers they control.

Moreover, service providers may wish to have multiple certificates with the same scope of authority. For example, a service provider with several regional gateway systems may want each system to be capable of signing for each of their numbers, but not want to have each system share the same private key.

The set of telephone numbers for which a particular certificate is valid is expressed in the certificate through a certificate extension; the certificate's extensibility mechanism is defined in RFC 5280 but the telephone number authorization extension is defined in this document.

3.2. Provisioning Private Keying Material

In order for authentication services to sign calls via the procedures described in [15], they must possess a private key corresponding to a certificate with authority over the calling number. This specification does not require that any particular entity sign requests, only that it be an entity with an appropriate private key; the authentication service role may be instantiated by any entity in a SIP network. For a certificate granting authority only over a particular number which has been issued to an end user, for example, an end user device might hold the private key and generate the signature. In the case of a service provider with authority over large blocks of numbers, an intermediary might hold the private key and sign calls.

The specification recommends distribution of private keys through PKCS#8 objects signed by a trusted entity, for example through the CMS package specified in [8].

4. Acquiring Credentials to Verify Signatures

This specification documents multiple ways that a verifier can gain access to the credentials needed to verify a request. As the validity of certificates does not depend on the circumstances of their acquisition, there is no need to standardize any single mechanism for this purpose. All entities that comply with [15] necessarily support SIP, and consequently SIP itself can serve as a way to acquire certificates. This specific does allow delivery through alternate means as well.

The simplest way for a verifier to acquire the certificate needed to verify a signature is for the certificate be conveyed along with the signature itself. In SIP, for example, a certificate could be carried in a multipart MIME body [9], and the URI in the Identity-Info header could specify that body with a CID URI [10]. However, in many environments this is not feasible due to message size restrictions or lack of necessary support for multipart MIME.

Alternatively, the Identity-Info header of a SIP request may contain a URI that the verifier dereferences with a network call. Implementations of this specification are required to support the use of SIP for this function (via the SUBSCRIBE/NOTIFY mechanism), as well as HTTP, via the Enrollment over Secure Transport mechanisms described in RFC 7030 [11].

A verifier can however have access to a service that grants access to certificates for a particular telephone number. Note however that there may be multiple valid certificates that can sign a call setup

request for a telephone number, and that as a consequence, there needs to be some discriminator that the signer uses to identify their credentials. The Identity-Info header itself can serve just such a discriminator.

4.1. Verifying Certificate Scope

The subjects of these certificates are the administrative entities to whom numbers are assigned or delegated. When a verifier is validating a caller's identity, local policy always determines the circumstances under which any particular subject may be trusted, but for the purpose of validating a caller's identity, this certificate extension establishes whether or not a signer is authorized to sign for a particular number.

The TN Authorization List certificate extension is identified by the following object identifier:

```
id-ce-TNAuthList OBJECT IDENTIFIER ::= { TBD }
```

The TN Authorization List certificate extension has the following syntax:

TNAuthorizationList ::= SEQUENCE SIZE (1..MAX) OF TNAuthorization

TNAuthorization ::= SEQUENCE SIZE (1..MAX) OF TNEntry

TNEntry ::= CHOICE {
 spid ServiceProviderIdentifierList,
 range TelephoneNumberRange,
 one E164Number }

ServiceProviderIdentifierList ::= SEQUENCE SIZE (1..3) OF
 OCTET STRING

-- When all three are present: SPID, Alt SPID, and Last Alt SPID

TelephoneNumberRange ::= SEQUENCE {
 start E164Number,
 count INTEGER }

E164Number ::= IA5String (SIZE (1..15)) (FROM ("0123456789"))

[TBD- do we really need to do IA5String? The alternative would be UTF8String, e.g.: UTF8String (SIZE (1..15)) (FROM ("0123456789"))]

The TN Authorization List certificate extension indicates the authorized phone numbers for the call setup signer. It indicates one or more blocks of telephone number entries that have been authorized for use by the call setup signer. There are three ways to identify the block: 1) a Service Provider Identifier (SPID) can be used to indirectly name all of the telephone numbers associated with that service provider, 2) telephone numbers can be listed in a range, and 3) a single telephone number can be listed.

Note that because large-scale service providers may want to associate many numbers, possibly millions of numbers, with a particular certificate, optimizations are required for those cases to prevent certificate size from becoming unmanageable. In these cases, the TN

Authorization List may be given by reference rather than by value, through the presence of a separate certificate extension that permits verifiers to either securely download the list of numbers associated with a certificate, or to verify that a single number is under the authority of this certificate. This optimization will be detailed in future version of this specification.

4.2. Certificate Freshness and Revocation

The problem of certificate freshness gains a new wrinkle in the telephone number context, because verifiers must establish not only that a certificate remains valid, but also that the certificate's scope contains the telephone number that the verifier is validating. Dynamic changes to number assignments can occur due to number portability, for example. So even if a verifier has a valid cached certificate for a telephone number (or a range containing the number), the verifier must determine that the entity that the signer is still a proper authority for that number.

This document therefore recommends the use of OCSP in high-volume environments for validating the freshness of certificates, per [12]. [TBD - depending on our algorithm choices this profile may need to be further profiled.]

5. Acknowledgments

Russ Housley, Brian Rosen, Cullen Jennings and Eric Rescorla provided key input to the discussions leading to this document.

6. IANA Considerations

This memo includes no request to IANA.

7. Security Considerations

This document is entirely about security. For further information on certificate security and practices, see RFC 3280 [5], in particular its Security Considerations.

8. Informative References

- [1] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [2] Barnes, R., Kent, S., and E. Rescorla, "Further Key Words for Use in RFCs to Indicate Requirement Levels", RFC 6919, April 1 2013.

- [3] 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.
- [4] Rosenberg, J. and H. Schulzrinne, "Session Initiation Protocol (SIP): Locating SIP Servers", RFC 3263, June 2002.
- [5] Housley, R., Polk, W., Ford, W., and D. Solo, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 3280, April 2002.
- [6] Rescorla, E., "HTTP Over TLS", RFC 2818, May 2000.
- [7] Cooper, D., Santesson, S., Farrell, S., Boeyen, S., Housley, R., and W. Polk, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 5280, May 2008.
- [8] Turner, S., "Asymmetric Key Packages", RFC 5958, August 2010.
- [9] Freed, N. and N. Borenstein, "Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types", RFC 2046, November 1996.
- [10] Levinson, E., "Content-ID and Message-ID Uniform Resource Locators", RFC 2392, August 1998.
- [11] Pritikin, M., Yee, P., and D. Harkins, "Enrollment over Secure Transport", RFC 7030, October 2013.
- [12] Deacon, A. and R. Hurst, "The Lightweight Online Certificate Status Protocol (OCSP) Profile for High-Volume Environments", RFC 5019, September 2007.
- [13] Peterson, J., Schulzrinne, H., and H. Tschofenig, "Secure Telephone Identity Problem Statement", draft-ietf-stir-problem-statement-03 (work in progress), January 2014.
- [14] Peterson, J., "Retargeting and Security in SIP: A Framework and Requirements", draft-peterson-sipping-retarget-00 (work in progress), February 2005.

- [15] Peterson, J., Jennings, C., and E. Rescorla,
"Authenticated Identity Management in the Session
Initiation Protocol (SIP)", draft-jennings-stir-
rfc4474bis-00 (work in progress), October 2013.

Authors' Addresses

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

Email: jon.peterson@neustar.biz

Sean Turner
IECA, Inc.
3057 Nutley Street, Suite 106
Farifax, VA 22031
US

Email: turners@ieca.com