Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: January 7, 2016

J. Peterson
NeuStar
C. Jennings
Cisco
E. Rescorla
RTFM, Inc.
July 6, 2015

Authenticated Identity Management in the Session Initiation Protocol (SIP)

draft-ietf-stir-rfc4474bis-04.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

Copyright Notice

Table of Contents

1. Introduction

This document provides enhancements to the existing mechanisms for authenticated identity management in the Session Initiation Protocol (SIP, [RFC3261]). 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 [RFC3966] or as the user portion of a SIP URI.

[RFC3261] stipulates several places within a SIP request where users can express an identity for themselves, primarily 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. This leaves SIP vulnerable to a category of abuses, including impersonation attacks that enable robocalling and related problems as described in [RFC7340].

[RFC3261] 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. A cryptographic approach, like the one described in this document, can provide a much stronger and less spoofable assurance of identity than the Caller ID services that the telephone network provides today.

[RFC4474] previously specified a means of signing portions of SIP requests in order to provide that identity assurance. However, RFC 4474 was in several ways misaligned with deployment realities (see [I-D.rosenberg-sip-rfc4474-concerns]). Most significantly, RFC 4474 did not deal well with telephone numbers as identifiers, despite their enduring use in SIP deployments. RFC 4474 also provided a signature over material that intermediaries in the field commonly altered. This specification therefore revises RFC 4474 in light of recent reconsideration of the problem space to align with the threat model in [RFC7375].

2. Background

The secure operation of many SIP applications and services depends on authorization policies. These policies may be automated, or they may be exercised 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 for detecting attempted 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 for authorizing the use of names in the example.com domain. Therefore, for the purposes of this specification, the credentials needed to prove a user is authorized to use a particular From header field 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. It relies on the authentication service adding to requests a SIP header, the Identity header, which contains that cryptographic assurance. In order to assist in the validation of the Identity header, 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 (see [RFC4916]).

This specification allows either a user agent or a proxy server to provide the authentication service function and/or 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 authentication services. However, for some deployments 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 prohobitively complex and require quite a good deal of additional endpoint behavior. Managing several credentials for the various devices could also be burdensome. In these cases, implementation the authentication service at an intermediary may be more practical. 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 a case where Alice, who has the home proxy of example.com and the address-of-record sip:alice@example.com, wants to communicate with Bob at sip:bob@example.org. They have no prior relationship, and Bob implements best practices to prevent impersonation attacks.

Alice generates an INVITE and places her identity, in this case her address-of-record, in the From header field of the request. She then sends an INVITE over TLS to an authentication service proxy for the example.com domain.

The authentication service authenticates Alice (possibly by sending a Digest authentication challenge) and validates that she is authorized to assert the identity that she populated in the From header field. This value is Alice's AoR, but in other cases it could be some different value that the proxy server has authority over, such as a telephone number. The proxy 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 for the identity (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 the example.com domain, is asserting that the originator of this request has been authenticated and that she is authorized to claim the identity 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 (a public key), 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). As the request has been validated, it is rendered to Bob. If the validation was unsuccessful, some other treatment would be applied by the receiving domain.

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 by 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.

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 URI value from the addr-spec component of From header field; this URI 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 [RFC3261], Section 16.4). If the identity field uses the TEL URI scheme [RFC3966], 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. An authentication service proceeding with a signature over a telephone number MAY add the optional 'canon' parameter to the request as described in that section. If the authentication service is not authoritative for the identity in question, it SHOULD process and forward the request normally, but it MUST NOT follow the steps below to add an Identity header. An authentication service MUST NOT add an Identity header to a request that already has one.

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 authenticate the request with the authentication scheme used for registration in its domain (e.g., Digest authentication).

If the authentication service is instantiated by a SIP user agent, a user agent may authenticate its own user through any system-

specific means, perhaps simply by virtue of having physical access to the user agent.

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 could return a 403 response code. In this case, 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, so no normative guidance is given here. For more information, see Section 10.4.

Step 3:

An authentication service MUST add a Date header field to SIP requests if one is not already present. The authentication service MUST 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 sixty seconds will ensure that the request is unlikely to upset any verifiers. If the Date header contains a time different by more than one minute 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.

See Section 7 for information on how the Date header field assists verifiers.

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 are 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 MAY an identity-extension signature and add an Identity-Extension header to the request containing this signature. The Identity-Extension header is created by this specification, but the header field value is left undefined. Only implementations that extend this base mechanism MAY popular this header field and add this signature. See Section 8.

Step 6:

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 the authentication service's 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.

4.2. 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 canonicalization process described in Section 6.1.1. 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, which usually involves dereferencing the Identity-Info header. See Section 5.2 for more information on these procedures.

Step 3:

The verifier MUST validate 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. The Identity-Reliance header provides important protections for non-INVITE transactions (such as MESSAGE or NOTIFY), but verifiers MAY elect not to verify Identity-Reliance when it protects SDP. 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:

If the request contains an Identity-Extension header, then if verifier supports the extension specified in the Identity-Extension header field, it SHOULD verify any associated following the procedures specified in that extension. See Section 8. If a verifier determines that such a signature in the message does not correspond to the reconstructed digest-string, then a 438 'Invalid

Identity Header' response SHOULD be returned. If the verifier does not support the extension, then the contents of the Identity-Extension header may be ignored.

Step 6:

The verifier MUST must furthermore ensure that the value of the Date header meets local policy for freshness (usually, within sixty seconds) and that it falls within the validity period of the credential used to sign the Identity header. For more on the attacks this prevents, see Section 10.1.

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 domain names or telephone numbers. These credentials may represent authority over an entire domain (such as example.com) or potentially a set of domains enumerated by the credential. 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.

This specification could also be used with credentials that cover a single name or URI, such as alice@example.com or sip:alice@example.com. This would require a modification to authentication service behavior to operate on a whole URI rather than a domain name. Because this is not believed to be a pressing use case, this is deferred to future work, but implementors should note this as a possible future direction.

Exceptions to such authentication service policies arise for cases like anonymity; if the AoR asserted in the From header field uses a form like 'sip:anonymous@example.com' (see [RFC3323]), then the 'example.com' proxy might authenticate only 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 domain names and/or telephone numbers or number ranges. Dereferencing the Identity-Info header (as described in the next section) MUST be 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 other implementation-specific way of acquiring the needed certificates, such as an offline store of periodically-updated credentials. If the URI 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. Further consideration of how to cache credentials is deferred to the credential mechanisms.

5.3. Handling Identity-Info URIs

An Identity-Info header MUST contain a URI which dereferences to a resource that contains the public key components of the credential used by the authentication service to sign a request. 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 serves as 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 a range of telephone and a user agent belonging to Alice has acquired a credential for a single telephone number within that range. Either would be eligible to sign a SIP request for the number in question. Verification services however need a means to differentiate which one performed the signature. The Identity-Info header performs that function.

If the optional "canon" parameter is present, it contains the result of the number canonicalization process performed by the authentication service (see Section 6.1.1) on the identity in the From. This value is provided purely informationally as an optimization for the verification service. The verification service MAY compute its own canonicalization of the number and compare this to the value in the "canon" parameter before performing any cryptographic functions in order to ascertain whether or not the two ends agree on the canonical number form.

5.4. Credential Systems

This document makes no recommendation for the use of any specific credential system. Today, there are two primary credential systems in place for proving ownership of domain names: certificates (e.g., X.509 v3, see [RFC5280]) and the domain name system itself (e.g., DANE, see [RFC6698]). It is envisioned that either could be used in the SIP identity 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 [RFC2585]). The Identity-Info headers may use the DNS URL scheme (see [RFC4501]) to designate 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 [I-D.ietf-stir-certificates]. One based on the domain name system is given in [I-D.kaplan-stir-cider].

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 RFC 4474, 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 transaction (a minute would most likely suffice).

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.

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.

In order to work with any such authorities, authentication and verification services must be able to 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 [RFC3966]). 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.

6.1.1. Canonicalization Procedures

In order to determine whether or not the user portion of a SIP URI is a telephone number, authentication services and verification services must perform the following canonicalization procedure on any SIP URI they inspect which contains a wholly numeric user part. Note that the same procedures are followed for creating the canonical form of URIs found in both the From and To header field values.

First, implementations must assess if the user-portion of the URI constitutes a telephone number. In some environments, numbers will be explicitly labeled by the use of TEL URIs or the 'user=phone' parameter, or implicitly by the presence of the '+' indicator at the start of the user-portion. Absent these indications, if there are numbers present in the user-portion, implementations may also detect that the user-portion of the URI contains a telephone number by determining whether or not those numbers would be dialable or routable in the local environment -- bearing in mind that the telephone number may be a valid E.164

number, a nationally-specific number, or even a private branch exchange number.

Once an implementation has identified a telephone number, it must construct a number string. Implementations MUST drop any leading +'s, any internal dashes, parentheses or other non-numeric characters, excepting only the leading "#" or "*" keys used in some special service numbers (typically, these will appear only in the To header field value). This MUST result in an ASCII string limited to "#", "*" and digits without whitespace or visual separators.

Next, an implementation must assess if the number string is a valid, globally-routable number with a leading country code. If not, implementations SHOULD convert the number into E.164 format, adding a country code if necessary; this may involve transforming the number from a dial string (see [RFC3966]), removing any national or international dialing prefixes or performing similar procedures. It is only in the case that an implementation cannot determine how to convert the number to a globally-routable format that this step may be skipped.

In some cases, further transformations MAY be made in accordance with specific policies used within the local domain. For example, one domain may only use local number formatting and need to convert all To/From user portions to E.164 by prepending country-code and region code digits; another domain might prefix usernames with trunk-routing codes and need to remove the prefix. Also, in some networks, the P-Asserted-Identity header field value is used in lieu of the From header field to convey the telephone number of the sender of a request; while it is not envisioned that most of those networks would or should make use of the Identity mechanism described in this specification, where they do, local policy might therefore dictate that the canonical string derive from the P-Asserted-Identity header field rather than the From. In any case where local policy canonicalizes the number into a form different from how it appears in the From header field, the use of the "canon" parameter by authentication services is RECOMMENDED, but because "canon" itself could then divulge information about users or networks, implementers should be mindful of the guidelines in Section 9.

The resulting canonical number string will be used as input to the hash calculation during signing and verifying processes.

The ABNF of this number string is:

```
tn-spec = [ "#" / "*" ] 1*DIGIT
```

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. Practically, entities that perform the authentication service role will sometimes alter the telephone numbers that appear in the To and From header field values, converting them to this format (though note this is not a function that [RFC3261] permits proxy servers to perform). The authentication service MAY also add the result of the canonicalization process of the From header field value to the "canon" parameter of the Identity-Info header. If the result of the canonicalization of the From header field value 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.

6.2. 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 it 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 [RFC3263] 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 [RFC2818], Section 3.1. While RFC 2818 [RFC2818] 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 four SIP headers: Identity, Identity-Reliance, Identity- Info, and Identity-Extension. Each of these headers can appear only once in a SIP request; Identity-Reliance are Identity-Extension are OPTIONAL, while Identity and Identity-Info are REQUIRED for securing requests with this specification. The grammar for the first three headers is (following the ABNF [RFC4234] in RFC 3261 [RFC3261]):

```
Identity = "Identity" HCOLON signed-identity-digest
signed-identity-digest = LDQUOT *base64-char RDQUOT

Identity-Reliance = "Identity-Reliance" HCOLON signed-identity-reliance-
digest
signed-identity-reliance-digest = LDQUOT *base64-char RDQUOT

Identity-Info = "Identity-Info" HCOLON ident-info
                 *( SEMI ident-info-params )
ident-info = LAQUOT absoluteURI RAQUOT
ident-info-params = ident-info-alg / canonical-str / ident-info-extension
ident-info-alg = "alg" EQUAL token
canonical-str = "canon" EQUAL tn-spec
ident-info-extension = generic-param

base64-char = ALPHA / DIGIT / "/" / "+"
```

The grammar for the Identity-Extension header field is given in Section 8.

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.

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 6.1.1). Otherwise, the first slot contains the AoR of the UA sending the message as taken from 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 (again, see Section 6.1.1). 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 [RFC3261]. RFC 3261 specifies that the BNF for weekday and month is a choice amongst a set of tokens. The RFC 4234 [RFC4234] 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, if the request contains an SDP message body, and if that SDP contains one or more "a=fingerprint" attributes, the value(s) of the attributes if they differ. Each attribute value consists of all characters following the colon after "a=fingerprint" including the algorithm description and hexadecimal key representation, any whitespace, carriage returns, and "/" line break indicators. If multiple non-identical "a=fingerprint" attributes appear in an SDP body, then all non-identical attributes values MUST be concatenated, with no separating character, after sorting the values in alphanumeric order. If the SDP body contains no "a=fingerprint" attribute, the fifth element MUST be empty, containing no whitespace, resulting in a "||" in the signed-identity-digest.

Sixth, the Identity-Extension header field value, if there is an Identity-Extension header field in the request. If the message has no Identity-Extension header, then the sixth slot MUST be empty, containing no whitespace, resulting in a "||" in the signed-identity-digest. characters.

Seventh, the Identity-Reliance header field value, if there is an Identity-Reliance header field in the request. If the message has no body, or no Identity-Reliance header, then the seventh slot MUST 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 [RFC3893]. The precise formulation of this digest-string is, therefore (following the ABNF[RFC4234] in RFC 3261 [RFC3261]):

```
digest-string = ( addr-spec / tn-spec ) "|" ( addr-spec / tn-spec ) "|"
                Method "|" SIP-date "|" [ sdp-fingerprint ] "|"
                                [ signed-identity-extension-digest ] "|"
                                [ signed-identity-reliance-digest ]

sdp-fingerprint = byte-string
```

For the definition of 'tn-spec' see Section 6.1.1.

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 11.3 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 [RFC3370] and base64 encode the results as specified in RFC 3548 [RFC3548]. 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.

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.

8. Extensibility

As future requirements may warrant increasing the scope of the Identity mechanism, this specification defines an optional Identity-Extension header. Each extension header field value MUST consist of a right hand side identifying the extension, an equals sign, and then a left hand side consisting of a signature over an element in a SIP request.

Future specifications that define extensions to the Identity mechanism must explicitly designate which elements of a SIP request are to be signed, how a canonical string of those elements is generated by both the authentication service and the verifier, and the mechanism and algorithms used to generate the signature (it is RECOMMENDED that these follow the algorithm choice of this specification). Note that per verifier behavior in Section 4.2, verifying an extension is always optional. An authentication service cannot assume that verifiers will understand any given extension. Verifiers that do support an extension may then trigger appropriate application-level behavior in the presence of a signature over additional part of the SIP request; authors of Identity extensions should provide appropriate extension-specific guidance to application developers on this point.

```
Identity-Extension = "Identity-Extension" HCOLON identity-extension-string
identity-extension-string = identity-extension-name EQUAL *base64-char
identity-extension-name = token
    signed-identity-extension-digest = LDQUOT *base64-char RDQUOT
```

Defining a new Identity-Extension requires a Standards Action; see Section 11.4.

No provision is made in this specification for multiple extensions to appear in a single SIP request.

9. Privacy Considerations

The purpose of this mechanism is to provide a strong identification of the originator of a SIP request, specifically a cryptographic assurance that the URI given in the From header field value can legitimately be claimed by the originator. This URI may contain a variety of personally identifying information, including the name of a human being, their place of work or service provider, and possibly further details. The intrinsic privacy risks associated with that URI are, however, no different from those of baseline SIP. Per the guidance in [RFC6973], implementors should make users aware of the privacy trade-off of providing secure identity.

The identity mechanism presented in this document is compatible with the standard SIP practices for privacy described in [RFC3323]. 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 even when 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 for that domain are meaningless.

[RFC3325] 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.

The optional "canon" parameter of the Identity-Info header specified in this document provides a canonicalized form of the telephone number of the originator of a call. In some contexts, local policy may be used to populate a "canon" that may differ substantially from the original From header field. Depending on those policies, potentially the "canon" parameter might divulge information about the originating network or user that might not appear elsewhere in the SIP request. Were it to be used to reflect the contents of the P-Asserted-Identity header field, for example, then "canon" would need to be removed when the P-Asserted-Identity header is removed to avoid any such leakage outside of a trust domain. Since, in those contexts, the canonical form of the sender's identity could not be reassembled by a verifier, and thus the Identity signature validation process would fail, using P-Asserted-Identity with the Identity "canon" parameter in this fashion is NOT RECOMMENDED outside of environments where SIP requests will never leave the trust domain.

Finally, note that unlike [RFC3325], 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 [RFC3261] 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 [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 60 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 transaction (for example, a re-INVITE) 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 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.

When signing a request that contains a fingerprint of keying material in SDP for DTLS-SRTP [RFC5763], this mechanism always provides a signature over that fingerprint. This signature prevents certain classes of impersonation attacks in which an attacker forwards or cut-and-pastes a legitimate request: although the target of the attack may accept the request, the attacker will be unable to exchange media with the target as they will not possess a key corresponding to the fingerprint. For example there are 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 a fingerprint for DTLS-SRTP. While this is a different form of interpretation than is commonly needed for robocalling, ultimately 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 subsequent communication. This signature also, per [RFC7258], reduces the potential for passive monitoring attacks against the SIP media. In environments where DTLS-SRTP is unsupported, however, this mechanism is not exercised and no protections are provided.

This mechanism also provides an optional full 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. 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.

10.1.1. Protection of the To Header and Retargeting

The mechanism in this document provides a signature over the identity information in the To header field value of requests. This provides a means for verifiers to detect replay attacks where a signed request originally sent to one target is modified and then forwarded by an attacker to another, unrelated target. Armed with the original value of the To header field, the recipient of a request may compare it to their own identity in order to determine whether or not the identity information in this call might have been replayed. However, any request may be legitimately retargeted as well, and as a result legitimate requests may reach a SIP endpoint whose user is not identified by the URI designated in the To header field value. It is therefore difficult for any verifier to decide whether or not some prior retargeting was "legitimate." Retargeting can also cause confusion when identity information is provided for requests sent in the backwards in a dialog, as the dialog identifiers may not match credentials held by the ultimate target of the dialog. For further information on the problems of response identity see [I-D.peterson-sipping-retarget].

Any means for authentication services or verifiers to anticipate retargeting 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 (see [RFC4916]); 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. And per the threat model of [RFC7375], resolving problems with 'connected' identity has little bearing on detecting robocalling or related impersonation attacks.

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 the 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

This document relies on the headers and response codes defined in RFC 4474. It also retains the requirements for the specification of new algorithms or headers related to the mechanisms described in that document.

11.1. Header Field Names

This document specifies a new SIP header called Identity-Reliance. Its syntax is given in Section 7. This header is 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-Reliance
Compact Form: N/A

This document also specifies a new SIP header called Identity-Extension. Its syntax is given in Section 8. A registry for Identity-Extension names is defined in Section 11.4.

Header Name: Identity-Extension
Compact Form: N/A

11.2. Identity-Info Parameters

The IANA has already created a registry for Identity-Info header parameters. This specification defines a new value called "canon" as defined in Section 5.3.

11.3. Identity-Info Algorithm Parameter Values

The IANA has already 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.

RFC4474 defined the 'rsa-sha1' value for this registry. That value is hereby deprecated, and should be treated as such. It is not believed that any implementations are making use of this value.

Future specifications may consider elliptical curves for smaller key sizes.

11.4. Identity-Extension Names

This specification requests that the IANA create a new registry for Identity-Extension names. The registry will consist solely of a list of names mapped to the Standards Track RFCs in which those extensions are defined.

The syntax of Identity-Extension names is given in Section 8. Registering a new Identity-Extension name requires a Standards Action.

This specification does not provide any initial values for Identity-Extension names.

12. Acknowledgments

The authors would like to thank Stephen Kent, Brian Rosen, Alex Bobotek, Paul Kyzviat, Jonathan Lennox, Richard Shockey, Martin Dolly, Andrew Allen, Hadriel Kaplan, Sanjay Mishra, Anton Baskov, Pierce Gorman, David Schwartz, Philippe Fouquart, Michael Hamer, Henning Schulzrinne, and Richard Barnes for their comments.

13. Changes from RFC4474

The following are salient changes from the original RFC 4474:

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 any DTLS-SRTP fingerprint in SDP as a mandatory element of the digest-string

Added the Identity-Reliance header

Added the Identity-Extension header and extensibility mechanism

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

14. References

14.1. Normative References

[RFC2818] Rescorla, E., "HTTP Over TLS", RFC 2818, May 2000.

[RFC3261] 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.

[RFC3263] Rosenberg, J. and H. Schulzrinne, "Session Initiation Protocol (SIP): Locating SIP Servers", RFC 3263, June 2002.

[RFC3280] 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.

[RFC3370] Housley, R., "Cryptographic Message Syntax (CMS) Algorithms", RFC 3370, August 2002.

[RFC3966] Schulzrinne, H., "The tel URI for Telephone Numbers", RFC 3966, December 2004.

[RFC5280] 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.

14.2. Informative References

[I-D.ietf-stir-certificates]
Peterson, J., "Secure Telephone Identity Credentials: Certificates", draft-ietf-stir-certificates-01 (work in progress), March 2015.

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

[I-D.peterson-sipping-retarget]
Peterson, J., "Retargeting and Security in SIP: A Framework and Requirements", draft-peterson-sipping-retarget-00 (work in progress), February 2005.

[I-D.rosenberg-sip-rfc4474-concerns]
Rosenberg, J., "Concerns around the Applicability of RFC 4474", draft-rosenberg-sip-rfc4474-concerns-00 (work in progress), February 2008.

[RFC2585] Housley, R. and P. Hoffman, "Internet X.509 Public Key Infrastructure Operational Protocols: FTP and HTTP", RFC 2585, May 1999.

[RFC3323] Peterson, J., "A Privacy Mechanism for the Session Initiation Protocol (SIP)", RFC 3323, November 2002.

[RFC3325] 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.

[RFC3548] Josefsson, S., "The Base16, Base32, and Base64 Data Encodings", RFC 3548, July 2003.

[RFC3893] Peterson, J., "Session Initiation Protocol (SIP) Authenticated Identity Body (AIB) Format", RFC 3893, September 2004.

[RFC4234] Crocker, D., Ed. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", RFC 4234, October 2005.

[RFC4474] Peterson, J. and C. Jennings, "Enhancements for Authenticated Identity Management in the Session Initiation Protocol (SIP)", RFC 4474, August 2006.

[RFC4501] Josefsson, S., "Domain Name System Uniform Resource Identifiers", RFC 4501, May 2006.

[RFC4916] Elwell, J., "Connected Identity in the Session Initiation Protocol (SIP)", RFC 4916, June 2007.

[RFC5763] Fischl, J., Tschofenig, H., and E. Rescorla, "Framework for Establishing a Secure Real-time Transport Protocol (SRTP) Security Context Using Datagram Transport Layer Security (DTLS)", RFC 5763, May 2010.

[RFC6698] Hoffman, P. and J. Schlyter, "The DNS-Based Authentication of Named Entities (DANE) Transport Layer Security (TLS) Protocol: TLSA", RFC 6698, August 2012.

[RFC6973] Cooper, A., Tschofenig, H., Aboba, B., Peterson, J., Morris, J., Hansen, M., and R. Smith, "Privacy Considerations for Internet Protocols", RFC 6973, July 2013.

[RFC7258] Farrell, S. and H. Tschofenig, "Pervasive Monitoring Is an Attack", BCP 188, RFC 7258, May 2014.

[RFC7340] Peterson, J., Schulzrinne, H., and H. Tschofenig, "Secure Telephone Identity Problem Statement and Requirements", RFC 7340, September 2014.

[RFC7375] Peterson, J., "Secure Telephone Identity Threat Model", RFC 7375, October 2014.

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