

Network Working Group  
Internet-Draft  
Intended status: Standards Track  
Expires: January 6, 2016

J. Peterson  
Neustar  
S. Turner  
IECA  
July 5, 2015

Secure Telephone Identity Credentials: Certificates  
draft-ietf-stir-certificates-02.txt

Abstract

In order to prove ownership of telephone numbers on the Internet, some kind of public infrastructure 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 January 6, 2016.

Copyright Notice

Copyright (c) 2015 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
4.2.1. Choosing a Verification Method . . . . .	8
4.2.2. Using OCSP with STIR Certificates . . . . .	9
4.2.2.1. OCSP Extension Specification . . . . .	10
4.2.3. Acquiring TN Lists By Reference . . . . .	11
5. Acknowledgments . . . . .	12
6. IANA Considerations . . . . .	12
7. Security Considerations . . . . .	12
8. Informative References . . . . .	12
Appendix A. ASN.1 Module . . . . .	14
Authors' Addresses . . . . .	15

## 1. Introduction

As is discussed in the STIR problem statement [I-D.ietf-stir-problem-statement], 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 [RFC5280]. 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 [I-D.ietf-stir-rfc4474bis]. An authentication service must be operated by an entity enrolled with the certification 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.

This document attempts to specify only the basic elements necessary for this architecture. Only through deployment experience will it be possible to decide directions for future work.

## 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 [RFC2119] and RFC 6919 [RFC6919].

## 3. Enrollment and Authorization

This document assumes a threefold model for certificate enrollment.

The first enrollment model is one where the certification 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 certification 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 dereferenced 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 Public Key Infrastructures (PKIs) who use the name constraints extension [RFC5280]; 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 may address adding a level of assurance indication to certificates to differentiate those enrolled from proof-of-possession versus delegation.

[TBD] Future versions of this specification may also discuss methods of partial delegation, where certificate holders delegate only part of their authority. For example, individual assignees may want to delegate to a service authority 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.

[TBD - what if the subject is considered a privacy leak?]

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 [RFC5280] 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 [I-D.ietf-stir-rfc4474bis], 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 [RFC5958].

## 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 [I-D.ietf-stir-rfc4474bis] 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 [RFC2046], and the URI in the Identity-Info header could specify that body with a CID URI [RFC2392]. 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 [RFC7030].

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 as 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 Telephony Number (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 TNEEntry
```

```
TNEEntry ::= 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 signed is still a proper authority for that number.

To verify the status of the certificate, the verifier needs the certificate, which is included with the call, and then would need to either:

- Rely on short-lived certificates and not check the certificate's status, or

- Rely on status information from the authority

The tradeoff between short lived certificates and using status information is the former's burden is on the front end (i.e., enrollment) and the latter's burden is on the back end (i.e., verification). Both impact call setup time, but it is assumed that performing enrollment for each call is more of an impact than using status information. This document therefore recommends relying on status information.

##### 4.2.1. Choosing a Verification Method

There are three common certificate verification mechanisms employed by CAs:

- Certificate Revocation Lists (CRLs) [RFC5280]

- Online Certificate Status Protocol (OCSP) [RFC6960], and

- Server-based Certificate Validation Protocol (SCVP) [RFC5055].

When relying on status information, the verifier needs to obtain the status information - but before that can happen, the verifier needs to know where to locate it. Placing the location of the status information in the certificate makes the certificate larger, but it eases the client workload. The CRL Distribution Point certificate extension includes the location of the CRL and the Authority Information Access certificate extension includes the location of OCSRP and/or SCVP servers; both of these extensions are defined in [RFC5280]. In all cases, the status information location is provided in the form of an URI.

CRLs are an obviously attractive solution because they are supported by every CA. CRLs have a reputation of being quite large (10s of MBytes), because CAs maintain and issue one monolithic CRL with all of their revoked certificates, but CRLs do support a variety of mechanisms to scope the size of the CRLs based on revocation reasons (e.g., key compromise vs CA compromise), user certificates only, and CA certificates only as well as just operationally deciding to keep the CRLs small. However, scoping the CRL introduces other issues (i.e., does the RP have all of the CRL partitions).

CAs in the STIR architecture will likely all create CRLs for audit purposes, but it NOT RECOMMENDED that they be relying upon for status information. Instead, one of the two "online" options is preferred. Between the two, OCSRP is much more widely deployed and this document therefore recommends the use of OCSRP in high-volume environments for validating the freshness of certificates, based on [RFC6960], incorporating some (but not all) of the optimizations of [RFC5019].

#### 4.2.2. Using OCSRP with STIR Certificates

Certificates compliant with this specification therefore SHOULD include a URL pointing to an OCSRP service in the Authority Information Access (AIA) certificate extension, via the "id-ad-ocsp" accessMethod specified in [RFC5280]. Baseline OCSRP however supports only three possible response values: good, revoked, or unknown. With some extension, OCSRP would not indicate whether the certificate is authorized for a particular telephone number that the verifier is validating.

[TBD] What would happen in the unknown case? Can we profile OCSRP usage so that unknown is never returned for our extension?

At a high level, there are two ways that a client might pose this authorization question:

For this certificate, is the following number currently in its scope of validity?

What are all the telephone numbers associated with this certificate, or this certificate subject?

Only the former lends itself to piggybacking on the OCSP status mechanism; since the verifier is already asking an authority about the certificate's status, why not reuse that mechanism, instead of creating a new service that requires additional round trips? Like most PKIX-developed protocols, OCSP is extensible; OCSP supports request extensions (including sending multiple requests at once) and per-request extensions. It seems unlikely that the verifier will be requesting authorization checks on multiple telephone numbers in one request, so a per-request extension is what is needed.

[TBD] HVE OCSP requires SHA-1 be used as the hash algorithm, we're obviously going to change this to be SHA-256.

The requirement to consult OCSP in real time results in a network round-trip time of day, which is something to consider because it will add to the call setup time. OCSP server implementations commonly pre-generate responses, and to speed up HTTPS connections, servers often provide OCSP responses for each certificate in their hierarchy. If possible, both of these OCSP concepts should be adopted for use with STIR.

#### 4.2.2.1. OCSP Extension Specification

The extension mechanism for OCSP follows X.509 v3 certificate extensions, and thus requires an OID, a criticality flag, and ASN.1 syntax as defined by the OID. The criticality specified here is optional: per [RFC6960] Section 4.4, support for all OCSP extensions is optional. If the OCSP server does not understand the requested extension, it will still provide the baseline validation of the certificate itself. Moreover, in practical STIR deployments, the issuer of the certificate will set the accessLocation for the OCSP AIA extension to point to an OCSP service that supports this extension, so the risk of interoperability failure due to lack of support for this extension is minimal.

The OCSP TNQuery extension is included as one of the requestExtensions in requests. It may also appear in the responseExtensions. When an OCSP server includes a number in the responseExtensions, this informs the client that the certificate is still valid for the number that appears in the TNQuery extension field. If the TNQuery is absent from a response to a query containing a TNQuery in its requestExtensions, then the server is not able to validate that the number is still in the scope of authority of the certificate.

```
id-pkix-ocsp-stir-tn    OBJECT IDENTIFIER ::= { id-pkix-ocsp TBD }  
  
TNQuery ::= E164Number
```

Note that HVE OCSF profile [RFC5019] prohibits the use of per-request extensions. As it is anticipated that STIR will use OCSF in a high-volume environment, many of the optimizations recommended by HVE are desirable for the STIR environment. This document therefore uses these extensions in a baseline OCSF environment with some HVE optimizations. [More TBD]

Ideally, once a certificate has been acquired by a verifier, some sort of asynchronous mechanism could notify and update the verifier if the scope of the certificate changes so that verifiers could implement a cache. While not all possible categories of verifiers could implement such behavior, some sort of event-driven notification of certificate status is another potential subject of future work. One potential direction is that a future SIP SUBSCRIBE/NOTIFY-based accessMethod for AIA might be defined (which would also be applicable to the method described in the following section) by some future specification.

#### 4.2.3. Acquiring TN Lists By Reference

Acquiring a list of the telephone numbers associated with a certificate or its subject lends itself to an application-layer query/response interaction outside of OCSF, one which could be initiated through a separate URI included in the certificate. The AIA extension (see [RFC5280]) supports such a mechanism: it designates an OID to identify the accessMethod and an accessLocation, which would most likely be a URI. A verifier would then follow the URI to ascertain whether the list of TNs authorized for use by the caller.

HTTPS is the most obvious candidate for a protocol to be used for fetching the list of telephone number associated with a particular certificate. This document defines a new AIA accessMethod, called "id-ad-stir-tn", which uses the following AIA OID:

```
id-ad-stir-tn    OBJECT IDENTIFIER ::= { id-ad TBD }
```

When the "id-ad-stir-tn" accessMethod is used, the accessLocation MUST be an HTTPS URI. The document returned by dereferencing that URI will contain the complete TN Authorization List (see Section 4.1) for the certificate.

Delivering the entire list of telephone numbers associated with a particular certificate will divulge to STIR verifiers information about telephone numbers other than the one associated with the particular call that the verifier is checking. In some environments, where STIR verifiers handle a high volume of calls, maintaining an up-to-date and complete cache for the numbers associated with crucial certificate holders could give an important boost to performance.

## 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 document makes use of object identifiers for the TN Certificate Extension defined in Section 4.1, TN-HVE OCSP extension in Section 4.2.2.1, and the TN by reference AIA access descriptor defined in Section 4.2.3. It therefore requests that the IANA make the following assignments:

- TN Certificate Extension in the SMI Security for PKIX Certificate Extension registry: <http://www.iana.org/assignments/smi-numbers/smi-numbers.xhtml#smi-numbers-1.3.6.1.5.5.7.1>
- TN-HVE OCSP extension in the SMI Security for PKIX Online Certificate Status Protocol (OCSP) registry: <http://www.iana.org/assignments/smi-numbers/smi-numbers.xhtml#smi-numbers-1.3.6.1.5.5.7.48.1>
- TNS by reference access descriptor in the SMI Security for PKIX Access Descriptor registry: <http://www.iana.org/assignments/smi-numbers/smi-numbers.xhtml#smi-numbers-1.3.6.1.5.5.7.48>

## 7. Security Considerations

This document is entirely about security. For further information on certificate security and practices, see RFC 3280 [RFC3280], in particular its Security Considerations.

## 8. Informative References

- [I-D.ietf-stir-problem-statement]  
Peterson, J., Schulzrinne, H., and H. Tschofenig, "Secure Telephone Identity Problem Statement and Requirements", draft-ietf-stir-problem-statement-05 (work in progress), May 2014.

- [I-D.ietf-stir-rfc4474bis]  
Peterson, J., Jennings, C., and E. Rescorla,  
"Authenticated Identity Management in the Session  
Initiation Protocol (SIP)", draft-ietf-stir-rfc4474bis-03  
(work in progress), March 2015.
- [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.
- [RFC2046] Freed, N. and N. Borenstein, "Multipurpose Internet Mail  
Extensions (MIME) Part Two: Media Types", RFC 2046,  
November 1996.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate  
Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC2392] Levinson, E., "Content-ID and Message-ID Uniform Resource  
Locators", RFC 2392, August 1998.
- [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.
- [RFC5019] Deacon, A. and R. Hurst, "The Lightweight Online  
Certificate Status Protocol (OCSP) Profile for High-Volume  
Environments", RFC 5019, September 2007.
- [RFC5055] Freeman, T., Housley, R., Malpani, A., Cooper, D., and W.  
Polk, "Server-Based Certificate Validation Protocol  
(SCVP)", RFC 5055, December 2007.

- [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.
- [RFC5958] Turner, S., "Asymmetric Key Packages", RFC 5958, August 2010.
- [RFC6919] Barnes, R., Kent, S., and E. Rescorla, "Further Key Words for Use in RFCs to Indicate Requirement Levels", RFC 6919, April 2013.
- [RFC6960] Santesson, S., Myers, M., Ankney, R., Malpani, A., Galperin, S., and C. Adams, "X.509 Internet Public Key Infrastructure Online Certificate Status Protocol - OCSP", RFC 6960, June 2013.
- [RFC7030] Pritikin, M., Yee, P., and D. Harkins, "Enrollment over Secure Transport", RFC 7030, October 2013.
- [RFC7299] Housley, R., "Object Identifier Registry for the PKIX Working Group", RFC 7299, July 2014.
- [X.680] USC/Information Sciences Institute, "Information Technology - Abstract Syntax Notation One.", ITU-T X.680, ISO/IEC 8824-1:2002, 2002.
- [X.681] USC/Information Sciences Institute, "Information Technology - Abstract Syntax Notation One: Information Object Specification", ITU-T X.681, ISO/IEC 8824-2:2002, 2002.
- [X.682] USC/Information Sciences Institute, "Information Technology - Abstract Syntax Notation One: Constraint Specification", ITU-T X.682, ISO/IEC 8824-3:2002, 2002.
- [X.683] USC/Information Sciences Institute, "Information Technology - Abstract Syntax Notation One: Parameterization of ASN.1 Specifications", ITU-T X.683, ISO/IEC 8824-4:2002, 2002.

#### Appendix A. ASN.1 Module

This appendix provides the normative ASN.1 [X.680] definitions for the structures described in this specification using ASN.1, as defined in [X.680] through [X.683].

TBD

Authors' Addresses

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

Email: [jon.peterson@neustar.biz](mailto:jon.peterson@neustar.biz)

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

Email: [turners@ieca.com](mailto:turners@ieca.com)

Network Working Group  
Internet-Draft  
Intended status: Standards Track  
Expires: June 21, 2018

J. Peterson  
Neustar  
S. Turner  
sn3rd  
December 18, 2017

Secure Telephone Identity Credentials: Certificates  
draft-ietf-stir-certificates-18

Abstract

In order to prevent the impersonation of telephone numbers on the Internet, some kind of credential system needs to exist that cryptographically asserts authority over telephone numbers. This document describes the use of certificates in establishing authority over telephone numbers, as a component 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 June 21, 2018.

Copyright Notice

Copyright (c) 2017 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. Authority for Telephone Numbers in Certificates . . . . .	4
4. Certificate Usage with STIR . . . . .	5
5. Enrollment and Authorization Using the TN Authorization List . . . . .	6
5.1. Constraints on Signing PASSporTs . . . . .	8
5.2. Certificate Extension Scope and Structure . . . . .	8
6. Provisioning Private Keying Material . . . . .	9
7. Acquiring Credentials to Verify Signatures . . . . .	9
8. JWT Claim Constraints Syntax . . . . .	10
9. TN Authorization List Syntax . . . . .	11
10. Certificate Freshness and Revocation . . . . .	13
10.1. Acquiring the TN List by Reference . . . . .	14
11. IANA Considerations . . . . .	15
11.1. ASN.1 Registrations . . . . .	15
11.2. Media Type Registrations . . . . .	16
12. Security Considerations . . . . .	16
13. References . . . . .	17
13.1. Normative References . . . . .	17
13.2. Informative References . . . . .	19
Appendix A. ASN.1 Module . . . . .	20
Acknowledgments . . . . .	22
Authors' Addresses . . . . .	22

## 1. Introduction

The Secure Telephone Identity Revisited (STIR) problem statement [RFC7340] identifies the primary enabler of robocalling, vishing (voicemail hacking), swatting, and related attacks as the capability to impersonate a calling party number. The starkest examples of these attacks are cases where automated callees on the Public Switched Telephone Network (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; therefore, appearing to call 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 the implementation of credentials that identify the

parties who control telephone numbers. With these credentials, parties can assert that they are in fact authorized to use telephony numbers (TNs), and thus they distinguish themselves from impersonators unable to present such credentials. For that reason, the STIR threat model [RFC7375] stipulates that "The design of the credential system envisioned as a solution to these threats must, for example, limit the scope of the credentials issued to carriers or national authorities to those numbers that fall under their purview." This document describes credential systems for telephone numbers based on [X.509] version 3 certificates in accordance with [RFC5280]. While telephone numbers have long been part of the X.509 standard (X.509 supports arbitrary naming attributes to be included in a certificate; the telephoneNumber attribute was defined in the 1988 [X.520] specification), this document provides ways to determine authority more aligned with telephone network requirements, including extending X.509 with a Telephony Number Authorization List certificate extension, which binds certificates to asserted authority for particular telephone numbers or, potentially, telephone number blocks or ranges.

In the STIR in-band architecture specified in [RFC8224], two basic types of entities need access to these credentials: authentication services and verification services (or verifiers). An authentication service must be operated by an entity enrolled with the certification authority (CA) (see Section 5), whereas a verifier need only trust the trust anchor of the authority and also have a means to access and validate the public keys associated with these certificates. Although the guidance in this document is written with the STIR in-band architecture in mind, the credential system described in this document could be useful for other protocols that want to make use of certificates to assert authority over telephone numbers on the Internet.

This document specifies only the credential syntax and semantics necessary to support this architecture. It does not assume any particular CA or deployment environment. We anticipate that some deployment experience will be necessary to determine optimal operational models.

## 2. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

### 3. Authority for Telephone Numbers in Certificates

At a high level, this specification details two non-exclusive approaches that can be employed to determine authority over telephone numbers with certificates.

The first approach is to leverage the existing subject of the certificate to ascertain that the holder of the certificate is authorized to claim authority over a telephone number. The subject might be represented as a domain name in the subjectAltName, such as an "example.net" where that domain is known to relying parties as a carrier, or represented with other identifiers related to the operation of the telephone network, including Service Provider Codes (SPCs) such as Operating Company Numbers (OCNs) or Service Provider Identifiers (SPIDs) via the TN Authorization List specified in this document. A relying party could then employ an external data set or service that determines whether or not a specific telephone number is under the authority of the carrier identified as the subject of the certificate and use that to ascertain whether or not the carrier should have authority over a telephone number. Potentially, a certificate extension to convey the URI of such an information service trusted by the issuer of the certificate could be developed (though this specification does not propose one). Alternatively, some relying parties could form bilateral or multilateral trust relationships with peer carriers, trusting one another's assertions just as telephone carriers in the Signaling System 7 (SS7) network today rely on transitive trust when displaying the calling party telephone number received through SS7 signaling.

The second approach is to extend the syntax of certificates to include a new attribute, defined here as the TN Authorization List, which contains a list of telephone numbers defining the scope of authority of the certificate. Relying parties, if they trust the issuer of the certificate as a source of authoritative information on telephone numbers, could therefore use the TN Authorization List instead of the subject of the certificate to make a decision about whether or not the signer has authority over a particular telephone number. The TN Authorization List could be provided in one of two ways: as a literal value in the certificate or as a network service that allows relying parties to query in real time to determine that a telephone number is in the scope of a certificate. Using the TN Authorization List rather than the certificate subject makes sense when, for example, for privacy reasons the certificate owner would prefer not to be identified, or in cases where the holder of the certificate does not participate in the sort of traditional carrier infrastructure that the first approach assumes.

The first approach requires little change to existing Public Key Infrastructure (PKI) certificates; for the second approach, we must define an appropriate enrollment and authorization process. For the purposes of STIR, the over-the-wire format specified in [RFC8224] accommodates either of these approaches: the methods for canonicalizing, for signing, for identifying and accessing the certificate, and so on remain the same; it is only the verifier behavior and authorization decision that will change, depending on the approach to telephone number authority taken by the certificate. For that reason, the two approaches are not mutually exclusive, and in fact a certificate issued to a traditional telephone network service provider could contain a TN Authorization List or not, were it supported by the CA issuing the credential. Regardless of which approach is used, certificates that assert authority over telephone numbers are subject to the ordinary operational procedures that govern certificate use per [RFC5280]. This means that verification services must be mindful of the need to ensure that they trust the trust anchor that issued the certificate and that they have some means to determine the freshness of the certificate (see Section 10).

#### 4. Certificate Usage with STIR

[RFC8224], Section 7.4 requires that all credential systems used by STIR explain how they address the requirements enumerated below. Certificates as described in this document address the STIR requirements as follows:

1. The URI [RFC3986] schemes permitted in the SIP Identity header "info" parameter, as well as any special procedures required to dereference the URIs: while normative text is given below in Section 7, this mechanism permits the HTTP [RFC7230], CID (Content-ID) [RFC2392], and SIP URI schemes to appear in the "info" parameter.
2. Procedures required to extract keying material from the resources designated by the URI: implementations perform no special procedures beyond dereferencing the "info" URI. See Section 7.
3. Procedures used by the verification service to determine the scope of the credential: this specification effectively proposes two methods, as outlined in Section 3: one where the subject (or, more properly, subjectAltName) of the certificate indicates the scope of authority through a domain name, and relying parties either trust the subject entirely or have some direct means of determining whether or not a number falls under a subject's authority; and another where an extension to the certificate as described in Section 9 identifies the scope of authority of the certificate.

4. The cryptographic algorithms required to validate the credentials: for this specification, that means the signature algorithms used to sign certificates. This specification REQUIRES that implementations support both the Elliptic Curve Digital Signature Algorithm (ECDSA) with the P-256 curve (see [DSS]) and RSA PKCS #1 v1.5 ("PKCS" stands for "Public-Key Cryptography Standards") (see [RFC8017], Section 8.2) for certificate signatures. Implementers are advised that the latter algorithm is mandated only as a transitional mechanism, due to its widespread use in existing PKIs, but we anticipate that this mechanism will eventually be deprecated.
5. Finally, note that all certificates compliant with this specification:
  - \* MUST provide cryptographic keying material sufficient to generate the ECDSA using P-256 and SHA-256 signatures necessary to support the ES256 hashed signatures required by PASSporT [RFC8225], which in turn follows the JSON Web Token (JWT) [RFC7519].
  - \* MUST support both ECDSA with P-256 and RSA PKCS #1 v1.5 for certificate signature verification.

This document also includes additional certificate-related requirements:

- o See Section 5.1 for requirements related to the JWT Claim Constraints certificate extension.
  - o See Section 7 for requirements related to relying parties acquiring credentials.
  - o See Sections 10 and 10.1 for requirements related to certificate freshness and the Authority Information Access (AIA) certificate extension.
5. Enrollment and Authorization Using the TN Authorization List

This document covers three models for enrollment when using the TN Authorization List extension.

The first enrollment model is one where the CA 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 Administration (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. This model requires top-down adoption of the model from regulators through to carriers. Assignees of E.164 numbering resources participating in this enrollment model should take appropriate steps to establish trust anchors.

The second enrollment model is a bottom-up approach where a CA requires that an entity prove control by means of some sort of test that, as with certification authorities for web PKI, might either be (1) automated or (2) a manual administrative process. As an example of an automated process, an authority might send a text message to a telephone number containing a URL (which might be dereferenced by the recipient) as a means of verifying that a user has control of a 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) might 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 a customer organization for a block of 100 numbers used by an IP PBX, 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 [RFC5280]; 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.

Future work might explore methods of partial delegation, where certificate holders delegate only part of their authority. For example, individual assignees may want to delegate to a service authority for text messages associated with their telephone number but not for other functions.

### 5.1. Constraints on Signing PASSportTs

The public key in the certificate is used to validate the signature on a JWT [RFC7519] that conforms to the conventions specified in PASSport [RFC8225]. This specification supports constraints on the JWT claims, thereby allowing the CA to grant different permissions to certificate holders -- for example, those enrolled from proof-of-possession versus delegation. A Certificate Policy (CP) and a Certification Practice Statement (CPS) [RFC3647] are produced as part of the normal PKI bootstrapping process (i.e., the CP is written first, and then the CA says how it conforms to the CP in the CPS). A CA that wishes to place constraints on the JWT claims MUST include the JWT Claim Constraints certificate extension in issued certificates. See Section 8 for information about the certificate extension.

### 5.2. Certificate Extension Scope and Structure

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 can be applied to 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 [RFC5280], but the TN Authorization List extension is specified in this document.

The subjects of certificates containing the TN Authorization List extension are typically the administrative entities to whom numbers are assigned or delegated. For example, a LEC might hold a certificate for a range of telephone numbers. In some cases, the organization or individual issued such a certificate may not want to associate themselves with a certificate; for example, a private individual with a certificate for a single telephone number might not want to distribute that certificate publicly if every verifier immediately knew their name. The certification authorities issuing certificates with the TN Authorization List extensions may, in accordance with their policies, obscure the identity of the subject,

though mechanisms for doing so are outside the scope of this document.

## 6. Provisioning Private Keying Material

In order for authentication services to sign calls via the procedures described in [RFC8224], they must hold a private key corresponding to a certificate with authority over the calling number. [RFC8224] does not require that any particular entity in a SIP deployment architecture 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 that 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 Cryptographic Message Syntax (CMS) package specified in [RFC5958].

## 7. 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 method of their acquisition, there is no need to standardize any single mechanism for this purpose. All entities that comply with [RFC8224] necessarily support SIP, and consequently SIP itself can serve as a way to deliver certificates. [RFC8224] provides an "info" parameter of the Identity header; this parameter contains a URI for the credential used to generate the Identity header. [RFC8224] also requires that documents that define credential systems list the URI schemes that may be present in the "info" parameter. For implementations compliant with this specification, three URI schemes are REQUIRED: the CID URI, the SIP URI, and the HTTP URI.

The simplest way for a verifier to acquire the certificate needed to verify a signature is for the certificate to be conveyed in a SIP request along with the signature itself. In SIP, for example, a certificate could be carried in a multipart MIME body [RFC2046], and the URI in the Identity header "info" parameter could specify that body with a CID URI [RFC2392]. However, in many environments this is not feasible due to message size restrictions or lack of necessary support for multipart MIME.

The Identity header "info" parameter in a SIP request may contain a URI that the verifier dereferences. Implementations of this specification are REQUIRED to support the use of SIP for this function (via the SUBSCRIBE/NOTIFY mechanism) as well as HTTP and HTTPS.

Note well that as an optimization, a verifier may have access to a service, a cache, or other local store that grants access to certificates for a particular telephone number. However, there may be multiple valid certificates that can sign a call setup request for a telephone number, and as a consequence, there needs to be some discriminator that the signer uses to identify their credentials. The Identity header "info" parameter itself can serve as such a discriminator, provided implementations use that parameter as a key when accessing certificates from caches or other sources.

#### 8. JWT Claim Constraints Syntax

Certificate subjects are limited to specific values for PASSporT claims with the JWT Claim Constraints certificate extension; issuers permit all claims by omitting the JWT Claim Constraints certificate extension from the certificate's extension field [RFC5280]. The extension is non-critical, applicable only to end-entity certificates, and defined with ASN.1 [X.680] [X.681] [X.682] [X.683] later in this section. The syntax of the claims is given in PASSporT; specifying new claims follows the procedures in [RFC8225], Section 8.3.

This certificate extension is optional, but if present, it constrains the claims that authentication services may include in the PASSporT objects they sign. Constraints are applied by issuers and enforced by verifiers when validating PASSporT claims as follows:

1. `mustInclude` indicates claims that MUST appear in the PASSporT in addition to `iat`, `orig`, and `dest`. The baseline claims of PASSporT ("`iat`", "`orig`", and "`dest`") are considered to be permitted by default and SHOULD NOT be included. If `mustInclude` is absent, `iat`, `orig`, and `dest` MUST appear in the PASSporT.
2. `permittedValues` indicates that if the claim name is present, the claim MUST contain one of the listed values.

Consider two examples with a PASSporT claim called "confidence" with values "low", "medium", and "high":

- o If a CA issues to an authentication service a certificate that contains the `mustInclude JWTClaimName "confidence"`, then an authentication service MUST include the "confidence" claim in all

PASSporTs it generates; a verification service will treat as invalid any PASSporT it receives with a PASSporT claim that does not include the "confidence" claim.

- o If a CA issues to an authentication service a certificate that contains the permittedValues JWTClaimName "confidence" and a permitted "high" value, then an authentication service will treat as invalid any PASSporT it receives with a PASSporT claim that does not include the "confidence" claim with a "high" value.

The JWT Claim Constraints certificate extension is identified by the following object identifier (OID), which is defined under the id-pe OID arc defined in [RFC5280] and managed by IANA (see Section 11):

```
id-pe-JWTClaimConstraints OBJECT IDENTIFIER ::= { id-pe 27 }
```

The JWT Claim Constraints certificate extension has the following syntax:

```
JWTClaimConstraints ::= SEQUENCE {
    mustInclude [0] JWTClaimNames OPTIONAL,
    -- The listed claim names MUST appear in the PASSporT
    -- in addition to iat, orig, and dest. If absent, iat, orig,
    -- and dest MUST appear in the PASSporT.
    permittedValues [1] JWTClaimPermittedValuesList OPTIONAL }
    -- If the claim name is present, the claim MUST contain one of
    -- the listed values.
( WITH COMPONENTS { ..., mustInclude PRESENT } |
  WITH COMPONENTS { ..., permittedValues PRESENT } )

JWTClaimPermittedValuesList ::= SEQUENCE SIZE (1..MAX) OF
    JWTClaimPermittedValues

JWTClaimPermittedValues ::= SEQUENCE {
    claim JWTClaimName,
    permitted SEQUENCE SIZE (1..MAX) OF UTF8String }

JWTClaimNames ::= SEQUENCE SIZE (1..MAX) OF JWTClaimName

JWTClaimName ::= IA5String
```

## 9. TN Authorization List Syntax

The subjects of certificates containing the TN Authorization List extension 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 the purpose of the

TN Authorization List extension in particular is to allow a verifier to ascertain when the CA has designated that the subject has authority over a particular telephone number or number range. The non-critical TN Authorization List certificate extension is included in the certificate's extension field [RFC5280]. The extension is defined with ASN.1 [X.680] [X.681] [X.682] [X.683]. The syntax and semantics of the extension are as follows.

The subjects of certificates containing the TN Authorization List extension are the administrative entities to whom numbers are assigned or delegated. In an end-entity certificate, the TN Authorization List indicates the TNs that it has authorized. In a CA certificate, the TN Authorization List limits the set of TNs for certification paths that include this certificate.

The TN Authorization List certificate extension is identified by the following object identifier (OID), which is defined under the id-pe OID arc defined in [RFC5280] and managed by IANA (see Section 11):

```
id-pe-TNAuthList OBJECT IDENTIFIER ::= { id-pe 26 }
```

The TN Authorization List certificate extension has the following syntax:

```
TNAuthorizationList ::= SEQUENCE SIZE (1..MAX) OF TNEEntry
```

```
TNEEntry ::= CHOICE {  
    spc    [0] ServiceProviderCode,  
    range [1] TelephoneNumberRange,  
    one    [2] TelephoneNumber  
}
```

```
ServiceProviderCode ::= IA5String
```

```
-- SPCs may be OCNs, various SPIDs, or other SP identifiers  
-- from the telephone network.
```

```
TelephoneNumberRange ::= SEQUENCE {  
    start TelephoneNumber,  
    count INTEGER (2..MAX),  
    ...  
}
```

```
TelephoneNumber ::= IA5String (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. SPCs as described in this document are a generic term for the identifiers used to designate service providers in telephone networks today. In North American context, these would include OCNs as specified in [ATIS-0300251], related SPIDs, or other similar identifiers for service providers. SPCs can be used to indirectly name all of the telephone numbers associated with that identifier for a service provider.
2. Telephone numbers can be listed in a range (in the TelephoneNumberRange format), which consists of a starting telephone number and then an integer count of numbers within the range, where the valid boundaries of ranges may vary according to national policies. The count field is only applicable to start fields' whose values do not include "\*" or "#" (i.e., a TelephoneNumber that does not include "\*" or "#"). count MUST NOT make the number increase in length (i.e., a TelephoneNumberRange with TelephoneNumber=10 with a count=91 is invalid); formally, given the inputs count and TelephoneNumber of length D TelephoneNumber + count MUST be less than  $10^D$ .
3. A single telephone number can be listed (as a TelephoneNumber).

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 the 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 (1) securely download the list of numbers associated with a certificate or (2) verify that a single number is under the authority of this certificate. For more on this optimization, see Section 10.1.

#### 10. Certificate Freshness and Revocation

Regardless of which of the approaches in Section 3 is followed for using certificates, a certificate verification mechanism is required. However, the traditional problem of certificate freshness gains a new wrinkle when using the TN Authorization List extension with telephone numbers or number ranges (as opposed to SPCs), 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 created the PASSporT, which includes a digital signature, is still a proper authority for that number.

To verify the status of such a certificate, the verifier needs to acquire the certificate if necessary (via the methods described in Section 7) and then would need to either:

- a. Rely on short-lived certificates and not check the certificate's status, or
- b. Rely on status information from the authority (e.g., the Online Certificate Status Protocol (OCSP)).

The trade-off between short-lived certificates and using status information is that the former's burden is on the front end (i.e., enrollment) and the latter's burden is on the back end (i.e., verification). Both impact call setup time, but some approaches to generating a short-lived certificate, like requiring one for each call, would incur a greater operational cost than acquiring status information. This document makes no particular recommendation for a means of determining certificate freshness for STIR, as this requires further study and implementation experience. Acquiring online status information for certificates has the potential to disclose private information [RFC7258] if proper precautions are not taken. Future specifications that define certificate freshness mechanisms for STIR MUST note any such risks and provide countermeasures where possible.

#### 10.1. Acquiring the TN List by Reference

One alternative to checking certificate status for a particular telephone number is simply acquiring the TN Authorization List by reference, that is, through dereferencing a URL in the certificate, rather than including the value of the TN Authorization List in the certificate itself.

Acquiring a list of the telephone numbers associated with a certificate or its subject lends itself to an application-layer query/response interaction outside of certificate status, one that could be initiated through a separate URI included in the certificate. The AIA extension (see [RFC5280]) supports such a mechanism: it designates an OID to identify the accessMethod and an accessLocation, which would most likely be a URI. A verifier would then follow the URI to ascertain whether the TNs in the list are authorized for use by the caller. As with the certificate extension defined in Section 9, a URI dereferenced from an end entity certificate will indicate the TNs which the caller has been authorized. Verifiers MUST support the AIA extension and the

dereferenced URI from a CA certificate limits the the set of TNs for certification paths that include this certificate.

HTTPS is the most obvious candidate for a protocol to be used for fetching the list of telephone numbers associated with a particular certificate. This document defines a new AIA accessMethod, called "id-ad-stirTNList", which uses the following AIA OID:

```
id-ad-stirTNList OBJECT IDENTIFIER ::= { id-ad 14 }
```

When the "id-ad-stirTNList" accessMethod is used, the accessLocation MUST be an HTTPS URI. Dereferencing the URI will return the complete DER encoded TN Authorization List (see Section 9) for the certificate with a Content-Type of application/tnauthlist (see Section 11.2).

Delivering the entire list of telephone numbers associated with a particular certificate will divulge to STIR verifiers information about telephone numbers other than the one associated with the particular call that the verifier is checking. In some environments, where STIR verifiers handle a high volume of calls, maintaining an up-to-date and complete cache for the numbers associated with crucial certificate holders could give an important boost to performance.

## 11. IANA Considerations

### 11.1. ASN.1 Registrations

This document makes use of object identifiers for the TN certificate extension defined in Section 9, the "TN List by reference" AIA access descriptor defined in Section 10.1, and the ASN.1 module identifier defined in Appendix A. Therefore, per this document, IANA has made the following assignments, as shown on <https://www.iana.org/assignments/smi-numbers>:

- o TN Authorization List certificate extension in the "SMI Security for PKIX Certificate Extension" (1.3.6.1.5.5.7.1) registry:

```
26 id-pe-TNAuthList
```

- o JWT Claim Constraints certificate extension in the "SMI Security for PKIX Certificate Extension" (1.3.6.1.5.5.7.1) registry:

```
27 id-pe-JWTClaimConstraints
```

- o TN List by reference access descriptor in the "SMI Security for PKIX Access Descriptor" (1.3.6.1.5.5.7.48) registry:

```
14 id-ad-stirTNList
```

- o The TN ASN.1 module in the "SMI Security for PKIX Module Identifier" (1.3.6.1.5.5.7.0) registry:

89 id-mod-tn-module

## 11.2. Media Type Registrations

Type name: application  
Subtype name: tnauthlist  
Required parameters: None  
Optional parameters: None  
Encoding considerations: Binary  
Security considerations: See Section 12 of [RFCTBD]  
Interoperability considerations:  
    The TN Authorization List inside this media type MUST be DER-encoded TNAuthorizationList.  
Published specification: [RFCTBD]  
Applications that use this media type:  
    Issuers and relying parties of secure telephone identity certificates, to limit the subject's authority to a particular telephone number or telephone number range.  
Fragment identifier considerations: None  
Additional information:  
    Deprecated alias names for this type: None  
    Magic number(s): None  
    File extension(s): None  
    Macintosh File Type Code(s): None  
Person & email address to contact for further information:  
    Jon Peterson <jon.peterson@team.neustar>  
Intended usage: COMMON  
Restrictions on usage: None  
Author: Sean Turner <sean@sn3rd.com>  
Change controller: The IESG <iesg@ietf.org>

[RFC editor's instruction: Please replace RFCTBD with the RFC number when this document is published.]

## 12. Security Considerations

This document is entirely about security. For further information on certificate security and practices, see [RFC5280], in particular its Security Considerations section.

If a certification authority issues a certificate attesting authority over many telephone numbers, the TNAuthList element can divulge to relying parties extraneous telephone numbers associated with the certificate which have no bearing on any given call in progress. The potential privacy risk can be exacerbated by the use of AIA, as

described in Section 10.1, to link many thousand of numbers to a single certificate. Even an SPC in a certificate can be used to link a certificate to a particular carrier and, with access to industry databases, potentially the set of numbers associated with that SPC. While these practices may not cause concern in some environments, in other scenarios alternative approaches could minimize the data revealed to relying parties. For example, a service provider with authority over a large block of numbers could generate short-lived certificates for individual TNs that are not so easily linked to the service provider or any other numbers that the service provider controls. Optimizations to facilitate acquiring short-lived certificates are a potential area of future work for STIR.

The TN Authorization List returned through a dereferenced URI is served over HTTPS; the TN Authorization List is therefore protected in transit. But, the TN Authorization List served is not a signed object and therefore the server is trusted to faithfully return the TN Authorization List provided to it by the list generator.

### 13. References

#### 13.1. Normative References

- [ATIS-0300251] ATIS Recommendation 0300251, "Codes for Identification of Service Providers for Information Exchange", 2007.
- [DSS] National Institute of Standards and Technology, U.S. Department of Commerce, "Digital Signature Standard (DSS)", NIST FIPS PUB 186-4, DOI 10.6028/NIST.FIPS.186-4, July 2013, <<http://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.186-4.pdf>>.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<https://www.rfc-editor.org/info/rfc2119>>.
- [RFC2392] Levinson, E., "Content-ID and Message-ID Uniform Resource Locators", RFC 2392, DOI 10.17487/RFC2392, August 1998, <<https://www.rfc-editor.org/info/rfc2392>>.
- [RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, RFC 3986, DOI 10.17487/RFC3986, January 2005, <<https://www.rfc-editor.org/info/rfc3986>>.

- [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, DOI 10.17487/RFC5280, May 2008, <<https://www.rfc-editor.org/info/rfc5280>>.
- [RFC5912] Hoffman, P. and J. Schaad, "New ASN.1 Modules for the Public Key Infrastructure Using X.509 (PKIX)", RFC 5912, DOI 10.17487/RFC5912, June 2010, <<https://www.rfc-editor.org/info/rfc5912>>.
- [RFC5958] Turner, S., "Asymmetric Key Packages", RFC 5958, DOI 10.17487/RFC5958, August 2010, <<https://www.rfc-editor.org/info/rfc5958>>.
- [RFC7230] Fielding, R., Ed. and J. Reschke, Ed., "Hypertext Transfer Protocol (HTTP/1.1): Message Syntax and Routing", RFC 7230, DOI 10.17487/RFC7230, June 2014, <<https://www.rfc-editor.org/info/rfc7230>>.
- [RFC7258] Farrell, S. and H. Tschofenig, "Pervasive Monitoring Is an Attack", BCP 188, RFC 7258, DOI 10.17487/RFC7258, May 2014, <<https://www.rfc-editor.org/info/rfc7258>>.
- [RFC7519] Jones, M., Bradley, J., and N. Sakimura, "JSON Web Token (JWT)", RFC 7519, DOI 10.17487/RFC7519, May 2015, <<https://www.rfc-editor.org/info/rfc7519>>.
- [RFC8017] Moriarty, K., Ed., Kaliski, B., Jonsson, J., and A. Rusch, "PKCS #1: RSA Cryptography Specifications Version 2.2", RFC 8017, DOI 10.17487/RFC8017, November 2016, <<https://www.rfc-editor.org/info/rfc8017>>.
- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <<https://www.rfc-editor.org/info/rfc8174>>.
- [RFC8224] Peterson, J., Jennings, C., Rescorla, E., and C. Wendt, "Authenticated Identity Management in the Session Initiation Protocol (SIP)", RFC 8224, DOI 10.17487/RFC8224, November 2017, <<https://www.rfc-editor.org/info/rfc8224>>.
- [RFC8225] Wendt, C. and J. Peterson, "PASSporT: Personal Assertion Token", RFC 8225, DOI 10.17487/RFC8225, November 2017, <<https://www.rfc-editor.org/info/rfc8225>>.

- [X.509] International Telecommunication Union, "Information technology - Open Systems Interconnection - The Directory: Public-key and attribute certificate frameworks", ITU-T Recommendation X.509, ISO/IEC 9594-8, October 2016, <<https://www.itu.int/rec/T-REC-X.509>>.
- [X.680] International Telecommunication Union, "Information Technology - Abstract Syntax Notation One (ASN.1): Specification of basic notation", ITU-T Recommendation X.680, ISO/IEC 8824-1, August 2015, <<https://www.itu.int/rec/T-REC-X.680>>.
- [X.681] International Telecommunication Union, "Information Technology - Abstract Syntax Notation One (ASN.1): Information object specification", ITU-T Recommendation X.681, ISO/IEC 8824-2, August 2015, <<https://www.itu.int/rec/T-REC-X.681>>.
- [X.682] International Telecommunication Union, "Information Technology - Abstract Syntax Notation One (ASN.1): Constraint specification", ITU-T Recommendation X.682, ISO/IEC 8824-3, August 2015, <<https://www.itu.int/rec/T-REC-X.682>>.
- [X.683] International Telecommunication Union, "Information Technology - Abstract Syntax Notation One (ASN.1): Parameterization of ASN.1 specifications", ITU-T Recommendation X.683, ISO/IEC 8824-4, August 2015, <<https://www.itu.int/rec/T-REC-X.683>>.

### 13.2. Informative References

- [RFC2046] Freed, N. and N. Borenstein, "Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types", RFC 2046, DOI 10.17487/RFC2046, November 1996, <<https://www.rfc-editor.org/info/rfc2046>>.
- [RFC3647] Chokhani, S., Ford, W., Sabett, R., Merrill, C., and S. Wu, "Internet X.509 Public Key Infrastructure Certificate Policy and Certification Practices Framework", RFC 3647, DOI 10.17487/RFC3647, November 2003, <<https://www.rfc-editor.org/info/rfc3647>>.
- [RFC7340] Peterson, J., Schulzrinne, H., and H. Tschofenig, "Secure Telephone Identity Problem Statement and Requirements", RFC 7340, DOI 10.17487/RFC7340, September 2014, <<https://www.rfc-editor.org/info/rfc7340>>.

- [RFC7375] Peterson, J., "Secure Telephone Identity Threat Model", RFC 7375, DOI 10.17487/RFC7375, October 2014, <<https://www.rfc-editor.org/info/rfc7375>>.
- [X.520] International Telecommunication Union, "Information technology - Open Systems Interconnection - The Directory: Selected attribute types", ITU-T Recommendation X.520, ISO/IEC 9594-6, October 2016, <<https://www.itu.int/rec/T-REC-X.520>>.

#### Appendix A. ASN.1 Module

This appendix provides the normative ASN.1 [X.680] definitions for the structures described in this specification using ASN.1, as defined in [X.680], [X.681], [X.682], and [X.683].

The modules defined in this document are compatible with the most current ASN.1 specifications published in 2015 (see [X.680], [X.681], [X.682], and [X.683]). None of the newly defined tokens in the 2008 ASN.1 (DATE, DATE-TIME, DURATION, NOT-A-NUMBER, OID-IRI, RELATIVE-OID-IRI, TIME, TIME-OF-DAY) are currently used in any of the ASN.1 specifications referred to here.

This ASN.1 module imports ASN.1 from [RFC5912].

```
TN-Module-2016
  { iso(1) identified-organization(3) dod(6) internet(1) security(5)
    mechanisms(5) pkix(7) id-mod(0) id-mod-tn-module(89) }

DEFINITIONS EXPLICIT TAGS ::= BEGIN

IMPORTS

id-ad, id-pe
FROM PKIX1Explicit-2009 -- From [RFC5912]
  { iso(1) identified-organization(3) dod(6) internet(1) security(5)
    mechanisms(5) pkix(7) id-mod(0) id-mod-pkix1-explicit-02(51) }

EXTENSION
FROM PKIX-CommonTypes-2009 -- From [RFC5912]
  { iso(1) identified-organization(3) dod(6) internet(1) security(5)
    mechanisms(5) pkix(7) id-mod(0) id-mod-pkixCommon-02(57) }

;

--
-- JWT Claim Constraints Certificate Extension
--
```

```
ext-jwtClaimConstraints EXTENSION ::= {
  SYNTAX JWTClaimConstraints IDENTIFIED BY id-pe-JWTClaimConstraints
}

id-pe-JWTClaimConstraints OBJECT IDENTIFIER ::= { id-pe 27 }

JWTClaimConstraints ::= SEQUENCE {
  mustInclude [0] JWTClaimNames OPTIONAL,
  -- The listed claim names MUST appear in the PASSporT
  -- in addition to iat, orig, and dest.  If absent, iat, orig,
  -- and dest MUST appear in the PASSporT.
  permittedValues [1] JWTClaimPermittedValuesList OPTIONAL }
( WITH COMPONENTS { ..., mustInclude PRESENT } |
  WITH COMPONENTS { ..., permittedValues PRESENT } )

JWTClaimPermittedValuesList ::= SEQUENCE SIZE (1..MAX) OF
  JWTClaimPermittedValues

JWTClaimPermittedValues ::= SEQUENCE {
  claim JWTClaimName,
  permitted SEQUENCE SIZE (1..MAX) OF UTF8String }

JWTClaimNames ::= SEQUENCE SIZE (1..MAX) OF JWTClaimName

JWTClaimName ::= IA5String

--
-- Telephony Number Authorization List Certificate Extension
--

ext-tnAuthList EXTENSION ::= {
  SYNTAX TNAuthorizationList IDENTIFIED BY id-pe-TNAuthList
}

id-pe-TNAuthList OBJECT IDENTIFIER ::= { id-pe 26 }

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

TNEntry ::= CHOICE {
  spc      [0] ServiceProviderCode,
  range    [1] TelephoneNumberRange,
  one      [2] TelephoneNumber
}

ServiceProviderCode ::= IA5String
```

```
-- SPCs may be OCNs, various SPIDs, or other SP identifiers
-- from the telephone network.

TelephoneNumberRange ::= SEQUENCE {
    start TelephoneNumber,
    count INTEGER (2..MAX),
    ...
}

TelephoneNumber ::= IA5String (SIZE (1..15)) (FROM ("0123456789#*"))

-- TN Access Descriptor

id-ad-stirTNList OBJECT IDENTIFIER ::= { id-ad 14 }

END
```

#### Acknowledgments

Anders Kristensen, Russ Housley, Brian Rosen, Cullen Jennings, Dave Crocker, Tony Rutkowski, John Braunberger, Eric Rescorla, and Martin Thomson provided key input to the discussions leading to this document. Russ Housley provided some direct assistance and text surrounding the ASN.1 module.

#### Authors' Addresses

Jon Peterson  
Neustar, Inc.

Email: jon.peterson@neustar.biz

Sean Turner  
sn3rd

Email: sean@sn3rd.com

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

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 January 7, 2016.

Copyright Notice

Copyright (c) 2015 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 . . . . .	4
3. Overview of Operations . . . . .	5
4. Signature Generation and Validation . . . . .	6
4.1. Authentication Service Behavior . . . . .	6
4.2. Verifier Behavior . . . . .	9
5. Credentials . . . . .	11
5.1. Credential Use by the Authentication Service . . . . .	11
5.2. Credential Use by the Verification Service . . . . .	12
5.3. Handling Identity-Info URIs . . . . .	13
5.4. Credential Systems . . . . .	13
6. Identity Types . . . . .	14
6.1. Telephone Numbers . . . . .	14
6.1.1. Canonicalization Procedures . . . . .	15
6.2. Domain Names . . . . .	17
7. Header Syntax . . . . .	18
8. Extensibility . . . . .	21
9. Privacy Considerations . . . . .	22
10. Security Considerations . . . . .	23
10.1. Handling of digest-string Elements . . . . .	23
10.1.1. Protection of the To Header and Retargeting . . . . .	26
10.2. Securing the Connection to the Authentication Service . . . . .	27
10.3. Authorization and Transitional Strategies . . . . .	28
10.4. Display-Names and Identity . . . . .	29
11. IANA Considerations . . . . .	29
11.1. Header Field Names . . . . .	29
11.2. Identity-Info Parameters . . . . .	30
11.3. Identity-Info Algorithm Parameter Values . . . . .	30
11.4. Identity-Extension Names . . . . .	30
12. Acknowledgments . . . . .	30
13. Changes from RFC4474 . . . . .	31
14. References . . . . .	31
14.1. Normative References . . . . .	31
14.2. Informative References . . . . .	32
Authors' Addresses . . . . .	33

## 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 prohibitively 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 add 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 add 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 and 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 replay 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 man-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.

#### 10.1.1.1. 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](mailto: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  
Obsoletes: 4474 (if approved)  
Intended status: Standards Track  
Expires: August 13, 2017

J. Peterson  
NeuStar  
C. Jennings  
Cisco  
E. Rescorla  
RTFM, Inc.  
C. Wendt  
Comcast  
February 9, 2017

Authenticated Identity Management in the Session Initiation Protocol  
(SIP)  
draft-ietf-stir-rfc4474bis-16.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 a SIP header field 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 13, 2017.

Copyright Notice

Copyright (c) 2017 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. Terminology . . . . .	4
3. Architectural Overview . . . . .	4
4. Identity Header Field Syntax . . . . .	6
4.1. PASSporT Construction . . . . .	7
4.1.1. Example Full and Compact Forms of PASSporT in Identity . . . . .	9
5. Example of Operations . . . . .	10
5.1. Example Identity Header Construction . . . . .	11
6. Signature Generation and Validation . . . . .	13
6.1. Authentication Service Behavior . . . . .	13
6.1.1. Handling Repairable Errors . . . . .	15
6.2. Verifier Behavior . . . . .	16
6.2.1. Authorization of Requests . . . . .	18
6.2.2. Failure Response Codes Sent by a Verification Service	18
6.2.3. Handling Retried Requests . . . . .	20
6.2.4. Handling the full form of PASSporT . . . . .	20
7. Credentials . . . . .	21
7.1. Credential Use by the Authentication Service . . . . .	21
7.2. Credential Use by the Verification Service . . . . .	22
7.3. 'info' parameter URIs . . . . .	23
7.4. Credential System Requirements . . . . .	23
8. Identity Types . . . . .	25
8.1. Differentiating Telephone Numbers from URIs . . . . .	25
8.2. Authority for Telephone Numbers . . . . .	26
8.3. Telephone Number Canonicalization Procedures . . . . .	26
8.4. Authority for Domain Names . . . . .	27
8.5. URI Normalization . . . . .	29
9. Extensibility . . . . .	30
10. Backwards Compatibility with RFC4474 . . . . .	30
11. Privacy Considerations . . . . .	31
12. Security Considerations . . . . .	33
12.1. Protected Request Fields . . . . .	33
12.1.1. Protection of the To Header and Retargeting . . . . .	35
12.2. Unprotected Request Fields . . . . .	35
12.3. Malicious Removal of Identity Headers . . . . .	36

12.4.	Securing the Connection to the Authentication Service	36
12.5.	Authorization and Transitional Strategies	37
12.6.	Display-Names and Identity	38
13.	IANA Considerations	39
13.1.	SIP Header Fields	39
13.2.	SIP Response Codes	39
13.3.	Identity-Info Parameters	39
13.4.	Identity-Info Algorithm Parameter Values	40
14.	Acknowledgments	40
15.	Changes from RFC4474	40
16.	References	40
16.1.	Normative References	41
16.2.	Informative References	42
	Authors' Addresses	44

## 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 canonical address-of-record (AoR) SIP URI employed to reach a user (such as 'sip:alice@atlanta.example.com'), or a telephone number, which commonly appears in either a TEL URI [RFC3966] or as the user portion of a SIP URI.

[RFC3261] specifies several places within a SIP request where users can express an identity for themselves, most prominently the user-populated From header field. However, in the absence of some sort of cryptographic authentication mechanism, the recipient of a SIP request has no way to verify that the From header field has been populated appropriately. This leaves SIP vulnerable to a category of abuses, including impersonation attacks that facilitate or enable robocalling, voicemail hacking, swatting, and related problems as described in [RFC7340]. Ideally, a cryptographic approach to identity can provide a much stronger and assurance of identity than the Caller ID services that the telephone network provides today, and one less vulnerable to spoofing.

[RFC3261] encourages user agents (UAs) to implement a number of potential authentication mechanisms, including Digest authentication, 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 for its part Digest authentication is limited by the fact that the originator and destination must share a prearranged secret. Practically speaking, originating user agents need to be able to securely communicate their

users' identity to destinations with which they have no previous association.

As an initial attempt to address this gap, [RFC4474] specified a means of signing portions of SIP requests in order to provide an identity assurance. However, RFC4474 was in several ways misaligned with deployment realities (see [I-D.rosenberg-sip-rfc4474-concerns]). Most significantly, RFC4474 did not deal well with telephone numbers as identifiers, despite their enduring use in SIP deployments. RFC4474 also provided a signature over material that intermediaries in existing deployments commonly altered. This specification therefore deprecates the RFC4474 syntax and behavior, reconsidering the problem space in light of the threat model in [RFC7375] and aligning the signature format with PASSporT [I-D.ietf-stir-passport]. Backwards compatibility considerations are given in Section 10.

## 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 [RFC2119].

In addition, this document uses three terms specific to the mechanism:

**Identity:** An identifier for the user of a communications service; for the purposes of SIP, either a SIP URI or a telephone number. Identities are derived from an "identity field" in a SIP request such as the From header field.

**Authentication Service:** A logical role played by a SIP entity that adds Identity headers to SIP requests.

**Verification Service (or "Verifier"):** A logical role played by a SIP entity that validates Identity headers in a SIP request.

## 3. Architectural Overview

The identity architecture for SIP defined in this specification depends on a logical "authentication service" which validates outgoing requests. An authentication service may be implemented either as part of a user agent or as a proxy server; typically, it is a component of a network intermediary like a proxy to which originating user agents send unsigned requests. Once the originator of the message has been authenticated, through pre-arranged means with the authentication service, the authentication service then creates and adds an Identity header field to the request. This

requires computing cryptographic information, including a digital signature over some components of messages, that lets other SIP entities verify that the sending user has been authenticated and its claim of a particular identity has been authorized. These "verification services" validate the signature and enable policy decisions to be made based on the results of the validation.

Policy decisions made after validation depend heavily on the verification service's trust for the credentials that the authentication service uses to sign requests. As robocalling, voicemail hacking, and swatting usually involve impersonation of telephone numbers, credentials that will be trusted by relying parties to sign for telephone numbers are a key component of the architecture. Authority over telephone numbers is, however, not as easy to establish on the Internet as authority over traditional domain names. This document assumes the existence of credentials for establishing authority over telephone numbers, for cases where the telephone number is the identity of the user, but this document does not mandate or specify a credential system; [I-D.ietf-stir-certificates] describes a credential system compatible with this architecture.

Although addressing the vulnerabilities in the STIR problem statement and threat model mostly requires dealing with telephone number as identities, SIP must also handle signing for SIP URIs as identities. This is typically easier to deal with, as these identities are issued by organizations that have authority over Internet domains. When a new user becomes associated with example.com, for example, the administrator of the SIP service for that domain can issue them an identity in that namespace, such as sip: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 other cases, Alice may herself be the owner of her own domain, 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 SIP as defined in [RFC3261], the necessary 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 that the domain owner has given the user agent the right to a name.

In order to share a cryptographic assurance of end-user SIP identity in an interdomain or intradomain context, an authentication service constructs tokens based on the PASSport [I-D.ietf-stir-passport] format, which is special encoding of a JSON [RFC7159] object

comprising values derived from certain header field values in the SIP request. The authentication service computes a signature over those JSON elements as PASSporT specifies. An encoding of the resulting PASSporT is then placed in the SIP Identity header field. In order to assist in the validation of the Identity header field, this specification also describes a parameter of the Identity header field 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 an identity assurance for SIP requests; solving this problem for SIP responses is outside the scope of this work (see [RFC4916]). Future work might specify ways that a SIP implementation could gateway PASSporTs to other protocols.

#### 4. Identity Header Field Syntax

The Identity and Identity-Info header fields that were previously defined in RFC4474 are here deprecated. This revised specification collapses the grammar of Identity-Info into the Identity header field via the "info" parameter. Note that unlike the prior specification in RFC4474, the Identity header field is now allowed to appear more than one time in a SIP request. The revised grammar for the Identity header field builds on the ABNF [RFC5234] in RFC 3261 [RFC3261] Section 25. It is as follows:

```
Identity = "Identity" HCOLON signed-identity-digest SEMI
          ident-info *( SEMI ident-info-params )
signed-identity-digest = 1*(base64-char / ".")
ident-info = "info" EQUAL ident-info-uri
ident-info-uri = LAQUOT absoluteURI RAQUOT
ident-info-params = ident-info-alg / ident-type /
                   ident-info-extension
ident-info-alg = "alg" EQUAL token
ident-type = "ppt" EQUAL token
ident-info-extension = generic-param

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

In addition to the "info" parameter, and the "alg" parameter previously defined in RFC4474, this specification defines the optional "ppt" parameter (PASSporT Type). The 'absoluteURI' portion of ident-info-uri MUST contain a URI; see Section 7.3 for more on choosing how to advertise credentials through this parameter.

The signed-identity-digest contains a base64 encoding of a PASSporT [I-D.ietf-stir-passport], which secures the request with a signature

that PASSporT generates over the JSON header and payload objects; some of those header and claim element values will mirror values of the SIP request.

#### 4.1. PASSporT Construction

For SIP implementations to populate the PASSporT header JSON object with fields from a SIP request, the following elements MUST be placed as the values corresponding to the designated JSON keys:

First, per baseline [I-D.ietf-stir-passport], the JSON "typ" key MUST have the value "passport".

Second, the JSON key "alg" MUST mirror the value of the optional "alg" parameter in the SIP Identity header field. Note if the "alg" parameter is absent from the Identity header, the default value is "ES256".

Third, the JSON key "x5u" MUST have a value equivalent to the quoted URI in the "info" parameter, per the simple string comparison rules of [RFC3986] section 6.2.1.

Fourth, if a PASSporT extension is in use, then the optional JSON key "ppt" MUST be present and have a value equivalent to the quoted value of the "ppt" parameter of the Identity header field.

An example of the PASSporT header JSON object without any extension is:

```
{ "typ": "passport",  
  "alg": "ES256",  
  "x5u": "https://www.example.com/cert.cer" }
```

To populate the PASSporT payload JSON object from a SIP request, the following elements MUST be placed as values corresponding to the designated JSON keys:

First, the JSON "orig" object MUST be populated. If the originating identity is a telephone number, then the array MUST be populated with a JSON object containing a "tn" element with a value set to the value of the quoted originating identity, a canonicalized telephone number (see Section 8.3). Otherwise, the object MUST be populated with a JSON object containing "uri" element, set to the value of the AoR of the UA sending the message as taken from the addr-spec of the From header field, per the procedures in Section 8.5.

Second, the JSON "dest" array MUST be populated. If the destination identity is a telephone number, then the array MUST be populated with a JSON object containing a "tn" element with a value set to the value of the quoted destination identity, a canonicalized telephone number (see Section 8.3). Otherwise, the array MUST be populated with a JSON object containing a "uri" element, set to the value of the addr-spec component of the To header field, which is the AoR to which the request is being sent, per the procedures in Section 8.5. Multiple JSON objects are permitted in "dest" for future compatibility reasons.

Third, the JSON key "iat" MUST appear. The authentication service SHOULD set the value of "iat" to an encoding of the value of the SIP Date header field as a JSON NumericDate (as UNIX time, per [RFC7519] Section 2), though an authentication service MAY set the value of "iat" to its own current clock time. If the authentication service uses its own clock time then the use of the full form of PASSporT is REQUIRED. In either case, the authentication service MUST NOT generate a PASSporT for a SIP request if the Date header is outside of its local policy for freshness (recommended sixty seconds).

Fourth, if the request contains an SDP message body, and if that SDP contains one or more "a=fingerprint" attributes, then the JSON key "mky" MUST appear with the algorithm(s) and value(s) of the fingerprint attributes (if they differ), following the format given in [I-D.ietf-stir-passport] Section 5.2.2.

For example:

```
{ "orig":{"tn":"12155551212"},  
  "dest":{"tn":"12155551213"},  
  "iat":1443208345 }
```

For information on the security properties of these SIP message elements, and why their inclusion mitigates replay attacks, see Section 12. Note that future extensions to PASSporT could introduce new claims, and that further SIP procedures could be required to extract information from the SIP request to populate the values of those claims; see Section 9 of this document.

The "orig" and "dest" arrays may contain identifiers of heterogeneous type; for example, the "orig" array might contain a "tn" claim, while the "dest" contains a "uri" claim. Also note that in some cases, the "dest" array may be populated with more than one value. This could for example occur when multiple "dest" identities are specified in a meshed conference. Defining how a SIP implementation would align



The compact form of the same PASSport object would appear in the Identity header as:

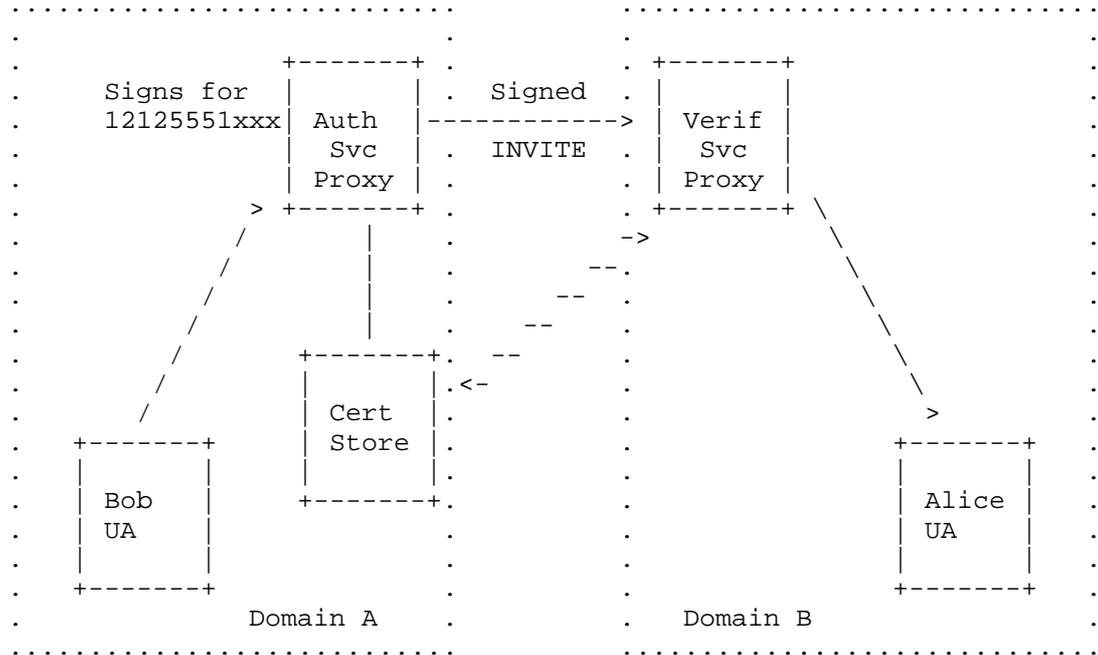
```
Identity: ..rq3pjTlhoRwakEGjHCnWSwUnshd0-zJ6F1VOgFWSjHBr8Qj \
pjlk-cpFYpFYsojNCpTzO3QfPOLckGaS6hEck7w; \
info=<https://biloxi.example.org/biloxi.cert>
```

5. Example of Operations

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

Imagine a case where Bob, who has the home proxy of example.com and the address-of-record sip:12155551212@example.com;user=phone, wants to communicate with Alice at sip:alice@example.org. They have no prior relationship, and Alice implements best practices to prevent impersonation attacks.

Bob's user agent generates an INVITE and places his address-of-record in the From header field of the request. He then sends an INVITE to an authentication service proxy for his domain.



The proxy authenticates Bob, and validates that he is authorized to assert the identity that he populated in the From header field. The proxy authentication service then constructs a PASSporT which contains a JSON representation of values which mirror certain parts of the SIP request, including the identity in the From header field value. As a part of generating the PASSporT, the authentication service signs a hash of that JSON header and payload with the private key associated with the appropriate credential for the identity (in this example, a certificate with authority to sign for numbers in a range from 12155551000 to 121555519999), and the signature is inserted by the proxy server into the Identity header field value of the request as a compact form of PASSporT. Alternatively, the JSON header and payload themselves might also have been included in the object when using the full form of PASSporT.

The proxy authentication service, as the holder of a private key with authority over Bob's telephone number, is asserting that the originator of this request has been authenticated and that he is authorized to claim the identity that appears in the From header field. The proxy inserts an "info" parameter into the Identity header field that tells Alice how to acquire keying material necessary to validate its credentials (a public key), in case she doesn't already have it.

When Alice's domain receives the request, a proxy verification service validates the signature provided in the Identity header field, and then determines that the authentication service credentials demonstrate authority over the identity in the From header field. This same validation operation might be performed by a verification service in Alice's user agent server. Ultimately, this valid request is rendered to Alice. If the validation were unsuccessful, some other treatment could be applied by the receiving domain or Alice's user agent.

#### 5.1. Example Identity Header Construction

For the following SIP request:

```

INVITE sip:bob@biloxi.example.org SIP/2.0
Via: SIP/2.0/TLS pc33.atlanta.example.com;branch=z9hG4bKnashds8
To: Alice <sip:alice@example.com>
From: Bob <sip:12155551212@example.com;user=phone>;tag=1928301774>
Call-ID: a84b4c76e66710
CSeq: 314159 INVITE
Max-Forwards: 70
Date: Fri, 25 Sep 2015 19:12:25 GMT
Contact: <sip:12155551212gateway.example.com>
Content-Type: application/sdp
Content-Length: 147

```

```

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

```

An authentication service will create a corresponding PASSporT object. The properly-serialized PASSporT header and payload JSON objects would look as follows. For the header, the values chosen by the authentication service at "example.org" might read:

```

{"alg":"ES256","typ":"passport","x5u":"https://cert.example.org/
passport.cer"}

```

The serialized payload will derive values from the SIP request (the From, To, and Date header field values) as follows:

```

{"dest":{"uri":["sip:alice@example.com"]},"iat":1443208345,
"orig":{"tn":"12155551212"}}

```

The authentication service would then generate the signature over the object following the procedures in [I-D.ietf-stir-passport] Section 6. That signature would look as follows:

```

rq3pjTlhoRwakeEGjHCnWSwUnshd0-zJ6F1VOgFWSjHBr8Qjplk-cpFYpFYs \
ojNCpTzO3QfPOLckGaS6hEck7w

```

An authentication service signing this request and using the compact form of PASSporT would thus generate and add to the request an Identity header field of the following form:

```

Identity: ..rq3pjTlhoRwakeEGjHCnWSwUnshd0-zJ6F1VOgFWSjHBr8Qjplk \
lk-cpFYpFYsojNCpTzO3QfPOLckGaS6hEck7w; \
info=<https://cert.example.org/passport.cer>

```

## 6. Signature Generation and Validation

SIP entities that instantiate the authentication service and verification service roles will, respectively, generate and validate the Identity header and the signature it contains.

### 6.1. Authentication Service Behavior

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 7.1). The authentication service role can be instantiated, for example, by an intermediary such as a proxy server or by a user agent. 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 proxy servers 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.

An authentication service adds the Identity header field to SIP requests. The procedures below define the steps that must be taken when each Identity header field is added. More than one Identity header field may appear in a single request, and an authentication service may add an Identity header field to a request that already contains one or more Identity header fields.

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

#### Step 1: Check Authority for the Identity

First, the authentication service must determine whether it is authoritative for the identity of the originator of the request. The authentication service extracts the identity from the URI value from the "identity field"; in ordinary operations, that is the addr-spec component of From header field. In order to determine whether the signature for the identity field should be over the entire identity field URI or just a telephone number, the authentication service MUST follow the process described in Section 8.1. That section will either lead to the telephone number canonicalization procedures in Section 8.3 for telephone numbers, or to the URI normalization procedures described in Section 8.5 for domain names. Whichever the result, if the authentication service is not authoritative for the

identity in question, it SHOULD process and forward the request normally unless the local policy is to block such requests. The authentication service MUST NOT add an Identity header field if the authentication service does not have the authority to make the claim it asserts.

#### Step 2: Authenticate the Originator

The authentication service MUST then determine whether or not the originator 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 originator 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 7.1 for more information.

Note that this check is performed only on the addr-spec in the identity field (e.g., the URI of the originator, like 'sip:alice@atlanta.example.com'); it does not cover the display-name portion of the From header field (e.g., 'Alice Atlanta'). For more information, see Section 12.6.

#### Step 3: Verify Date is Present and Valid

An authentication service MUST add a Date header field to SIP requests that do not have one. The authentication service MUST ensure that any preexisting Date header field 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 field value 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 field in order to cause a request to fail verification; the Identity header field is not intended to provide a

perfect record of when messages are processed. Finally, the authentication service MUST verify that both the Date header field and the current time fall within the validity period of its credential.

See Section 12.1 for information on how the Date header field assists verifiers.

#### Step 4: Populate and Add the Identity Header

Subsequently, the authentication service MUST form a PASSporT object and add a corresponding Identity header field to the request containing either the full or compact form of PASSporT. For the baseline PASSporT header (headers containing no "ppt" parameter), this follows the procedures in Section 4; if the authentication service is using an alternative "ppt" format, it MUST add an appropriate "ppt" parameter and follow the procedures associated with that extension (see Section 9). After the Identity header field has been added to the request, the authentication service MUST also add a "info" parameter to the Identity header field. The "info" parameter contains a URI from which the authentication service's credential can be acquired; see Section 7.3 for more on credential acquisition.

An authentication service MAY use the full form of the PASSporT in the Identity header field. The presence of the full form is OPTIONAL because the information carried in the baseline PASSporT headers and claims is usually redundant with information already carried elsewhere in the SIP request. Using the compact form can significantly reduce SIP message size, especially when the PASSporT payload contains media keys. The syntax of the compact form is given in [I-D.ietf-stir-passport] Section 7; essentially, it contains only the signature component of the PASSporT.

Note that per the behavior specified in [I-D.ietf-stir-passport], use of the full form is mandatory when optional extensions are included. See Section 9.

#### 6.1.1. Handling Repairable Errors

Also, in some cases, a request signed by an authentication service will be rejected by the verification service on the receiving side, and the authentication service will receive a SIP 4xx status code in the backwards direction, such as a 438 indicating a verification failure. If the authentication service did not originally send the full form of the PASSporT object in the Identity header field, it SHOULD retry the request with the full form after receiving a 438 response; however implementations SHOULD NOT retry the request more than once. Authentication services implemented at proxy servers

would retry such a request as a ssequential for, by re-processing the destination as a new target and handling it serially as described in Section 16.6 of [RFC3261].

The information in the full form is useful on the verification side for debugging errors, and there are some known causes of verification failures (such as the Date header field value changing in transit, see Section 12.1 for more information) that can be resolved by the inclusion of the full form of PASSporT.

Finally, the authentication service forwards the message normally.

## 6.2. 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 one or more Identity header fields, it inspects the signature(s) to verify the identity of the originator of the message. The results of a verification are provided as input to an authorization process that is outside the scope of this document.

A SIP request may contain zero, one, or more Identity header fields. A verification service performs the steps below on each Identity header field that appears in a request. If a verification service cannot use any Identity header in a request, due to the absence of Identity headers or unsupported "ppt" parameters, and the presence of an Identity header field 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 the backwards direction. For more on this and other verifier responses, see Section 6.2.2.

In order to verify an Identity header field in a message, an entity acting as a verifier MUST perform the following steps, in the order here specified. Note that when an Identity header field contains a full form PASSporT object, the verifier MUST follow the additional procedures in Section 6.2.4.

Step 1: Check for an Unsupported "ppt"

The verifier MUST inspect any optional "ppt" parameter appearing in the Identity header. If no "ppt" parameter is present, then the verifier proceeds normally below. If a "ppt" parameter value is present, and the verifier does not support it, it MUST ignore the Identity header field. If a supported "ppt" parameter value is present, the verifier proceeds with Step 2, and will ultimately follow the "ppt" variations described in Step 5.

#### Step 2: Determine the Originator's Identity

In order to determine whether the signature for the identity field should be over the entire identity field URI or just a telephone number, the verification service MUST follow the process described in Section 8.1. That section will either lead to the telephone number canonicalization procedures in Section 8.3 for telephone numbers, or to the URI normalization procedures described in Section 8.5 for domain names.

#### Step 3: Identify Credential for Validation

The verifier must ensure that it has access to the proper keying material to validate the signature in the Identity header field, which usually involves dereferencing a URI in the "info" parameter of the Identity header field. See Section 7.2 for more information on these procedures. If the verifier does not support the credential described in the "info" parameter, then it treats the credential for this header field as unsupported.

#### Step 4: Check the Freshness of Date

The verifier furthermore ensures that the value of the Date header field of the request meets local policy for freshness (sixty seconds is RECOMMENDED) and that it falls within the validity period of the credential used to sign the Identity header field. For more on the attacks this prevents, see Section 12.1. If the full form of the PASSporT is present, the verifier SHOULD compare the "iat" value in the PASSporT to the Date header field value in the request. If the two are different, and the "iat" value differs from the Date header field value but remains within verification service policy for freshness, the verification service SHOULD perform the computation required by Step 5 using the "iat" value instead of the Date header field value.

#### Step 5: Validate the Signature

The verifier MUST validate the signature in the Identity header field over the PASSporT object. For baseline PASSporT objects (with no Identity header field "ppt" parameter) the verifier MUST follow the procedures for generating the signature over a PASSporT object described in Section 4. If a "ppt" parameter is present (and per Step 1, is supported), the verifier follows the procedures for that "ppt" (see Section 9). If a verifier determines that the signature in the Identity does not correspond to the reconstructed signed-identity-digest, then the Identity header field should be considered invalid.

### 6.2.1. Authorization of Requests

The verification of an Identity header field does not entail any particular treatment of the request. The handling of the message after the verification process depends on how the verification service is implemented and on local policy. This specification does not propose any authorization policy for user agents or proxy servers to follow based on the presence of a valid Identity header field, the presence of an invalid Identity header field, or the absence of an Identity header field, or a stale Date header field value, but it is anticipated that local policies could involve making different forwarding decisions in intermediary implementations, or changing how the user is alerted, or how identity is rendered, in user agent implementations.

The presence of multiple Identity header fields within a message raises the prospect that a verification services could receive a message containing some valid and some invalid Identity header fields. As a guideline, this specification recommends that only if a verifier determines all Identity header fields within a message are invalid should the request be considered to have an invalid identity. If at least one Identity header field value is valid and from a trusted source, then relying parties can use that header for authorization decisions regardless of whether other untrusted or invalid Identity headers appear in a request.

### 6.2.2. Failure Response Codes Sent by a Verification Service

RFC4474 originally defined four response codes for failure conditions specific to the Identity header field and its original mechanism. These status codes are retained in this specification, with some slight modifications. Also, this specification details responding with 403 when a stale Date header field value is received.

A 428 response will be sent (per Section 6.2) when an Identity header field is required, but no Identity header field without a "ppt" parameter, or with a supported "ppt" value, has been received. In the case where one or more Identity header fields with unsupported "ppt" values have been received, then a verification service may send a 428 with a human-readable reason phrase like "Use Supported PASSporT Format". Note however that this specification gives no guidance on how a verification service might decide to require an Identity header field for a particular SIP request. Such authorization policies are outside the scope of this specification.

The 436 'Bad Identity Info' response code indicates an inability to acquire the credentials needed by the verification service for validating the signature in an Identity header field. Again, given

the potential presence of multiple Identity header fields, this response code should only be sent when the verification service is unable to deference the URIs and/or acquire the credentials associated with all Identity header fields in the request. This failure code could be repairable if the authentication service resends the request with an 'info' parameter pointing to a credential that the verification service can access.

The 437 'Unsupported Credential' is sent when a verification service can acquire, or already holds, the credential represented by the 'info' parameter of at least one Identity header field in the request, but does not support said credential(s), for reasons such as failing to trust the issuing CA, or failing to support the algorithm with which the credential was signed.

The 438 'Invalid Identity Header' response indicates that of the set of Identity header fields in a request, no header field with a valid and supported PASSporT object has been received. Like the 428 response, this is sent by a verification service when its local policy dictates that a broken signature in an Identity header field is grounds for rejecting a request. Note that in some cases, an Identity header field may be broken for other reasons than that an originator is attempting to spoof an identity: for example, when a transit network alters the Date header field of the request. Sending a full form PASSporT can repair some of these conditions (see Section 6.2.4), so the recommended way to attempt to repair this failure is to retry the request with the full form of PASSporT if it had originally been sent with the compact form. The alternative reason phrase 'Invalid PASSporT' can be used when an extended full form PASSporT lacks required headers or claims, or when an extended full form PASSporT signaled with the "ppt" parameter lacks required claims for that extension. Sending a string along these lines will help humans debugging the sending system.

Finally, a 403 response may be sent when the verification service receives a request with a Date header field value that is older than the local policy for freshness permits. The same response may be used when the "iat" in the full form of a PASSporT has a value older than the local policy for freshness permits. The reason phrase "Stale Date" can be sent to help humans debug the failure.

Future specifications may explore ways, including Reason codes or Warning headers, to communicate further information that could be used to disambiguate the source of errors in cases with multiple Identity headers in a single request, or provide similar detailed feedback for debugging purposes.

### 6.2.3. Handling Retried Requests

If a verification service sends a failure response in the backwards direction, the authentication service may retry the request as described in Section 6.1.1. If the authentication service is instantiated at a proxy server, then it will retry the request as a sequential fork. Verification services implemented at a proxy server will recognize this request as a spiral rather than a loop due to the proxy behavior fix documented in [RFC5393] Section 4.2. However, if the verification service is implemented in an endpoint, the endpoint will need to override the default UAS behavior (in particular, the SHOULD in [RFC3261] Section 8.2.2.2) to accept this request as a spiral rather than a loop.

### 6.2.4. Handling the full form of PASSporT

If the full form of PASSporT is present in an Identity header, this permits the use of optional extensions as described in [I-D.ietf-stir-passport] Section 8.3. Furthermore, the verification service can extract from the "orig" and "dest" elements of the PASSporT full form the canonical telephone numbers created by the authentication service, as well as an "iat" claim corresponding to the Date header field that the authentication service used. These values may be used to debug canonicalization problems, or to avoid unnecessary signature breakage caused by intermediaries that alter certain SIP header field values in transit.

However, the verification service MUST NOT treat the value in the "orig" of a full form PASSporT as the originating identity of the call: the originating identity of the call is always derived from the SIP signaling, and it is that value, per the procedures above in Section 6.2 Step 2, which is used to recompute the signature at the verification service. That value, rather than the value inside the PASSporT object, is rendered to an end user in ordinary SIP operations, and if a verification service were to simply trust that the value in the "orig" corresponded to the call that it received without comparing it to the call signaling, this would enable various cut-and-paste attacks. As an optimization, when the full form is present, the verification service MAY delay performing that cryptographic operation and first compute its own canonicalization of an originating telephone number to compare it to the values in the "orig" element of PASSporT. This would allow the verification service to ascertain whether or not the two ends agree on the canonical number form; if they do not, then surely the signature validation would fail.

## 7. Credentials

This section gives general guidance on the use of credential systems by authentication and verification services, as well as requirements that must be met by credential systems that conform with this architecture. It does not mandate any specific credential system.

Furthermore, this specification allows either a user agent or a proxy server to provide the authentication service function and/or the verification service function. For the purposes of end-to-end security, it is obviously preferable for end systems to acquire their own credentials; in this case 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. Synchronizing keying material across multiple devices may be prohibitively complex and require quite a good deal of additional endpoint behavior. Managing several credentials for the various devices could also be burdensome. Thus, for reasons of credential management alone, implementing the authentication service at an intermediary may be more practical. This trade-off needs to be understood by implementers of this specification.

### 7.1. Credential Use by the Authentication Service

In order to act as an authentication service, a SIP entity must possess the private keying material of one or more credentials that cover domain names or telephone numbers. These credentials may represent authority over one domain (such as example.com) or 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's authority is expressed is specific to the credential mechanism.

Authorization of the use of a particular username or telephone number in the From header field value 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 field 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 header field that do not correspond to the user-portion of From header fields, 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 the Proxy-Authorization header field 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 field. 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 implementers 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.

## 7.2. Credential Use by the Verification Service

In order to act as a verification service, a SIP entity must have a way to acquire credentials for authorities over particular domain names, telephone numbers and/or number ranges. Dereferencing the URI found in the "info" parameter of the Identity header field (as described Section 7.3) 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 "info" parameter, unless they have some implementation-specific way of acquiring the needed keying material, such as an offline store of periodically-updated credentials. The 436 'Bad Identity Info' response exists for cases where the verification service cannot deference the URI in the "info" parameter.

This specification does not propose any particular policy for a verification service to determine whether or not the holder of a credential is the appropriate party to sign for a given SIP identity. Guidance on this is deferred to credential mechanism specifications.

Verification service implementations supporting this specification may wish to 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 of authority, or the URI given in the "info" parameter value. Further consideration of how to cache credentials is deferred to the credential mechanism specifications.

### 7.3. 'info' parameter URIs

An "info" parameter 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 "info" parameter be dereferencable by any entity that could plausibly receive the request. For common cases, this means that the URI SHOULD be dereferencable by any entity on the public Internet. In constrained deployment environments, a service private to the environment MAY be used instead.

Beyond providing a means of accessing credentials for an identity, the "info" parameter 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 numbers 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 "info" parameter performs that function.

### 7.4. Credential System Requirements

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 "info" parameter could for example give an HTTP URL of the Content-Type 'application/pkix-cert' pointing to a certificate (following the conventions of [RFC2585]). The "info" parameter might 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], but this specification can work with other credential systems; for example, using the DNS was proposed 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 "info" parameter, 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 required to validate the credentials (e.g. for certificates, any algorithms used by certificate authorities to sign certificates themselves), and

- how the associated credentials will support the mandatory signing algorithm(s) required by PASSport [I-D.ietf-stir-passport].

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 header field "info" parameter 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 "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).

## 8. Identity Types

The problem statement of STIR [RFC7340] focuses primarily on cases where the called and calling parties identified in the To and From header field values use telephone numbers, as this remains the dominant use case in the deployment of SIP. However, the Identity header mechanism also works with SIP URIs without telephone numbers (of the form "sip:user@host"), and potentially other identifiers when SIP interworks with other protocols.

Authentication services confirm the identity of the originator of a call, which is typically found in the From header field value. The guidance in this specification also applies to extracting the URI containing the originator's identity from the P-Asserted-Identity header field value instead of the From header field value. In some trusted environments, the P-Asserted-Identity header field is used in lieu of the From header field to convey the address-of-record or telephone number of the originator of a request; where it does, local policy might therefore dictate that the canonical identity derives from the P-Asserted-Identity header field rather than the From header field.

Ultimately, in any case where local policy canonicalizes the identity into a form different from how it appears in the From header field, the use of the full form of PASSporT by authentication services is RECOMMENDED, but because the "orig" claim of PASSporT itself could then divulge information about users or networks, implementers should be mindful of the guidelines in Section 11.

### 8.1. Differentiating Telephone Numbers from URIs

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 procedure on any SIP URI they inspect which contains a numeric user part. Note that the same procedures are followed for creating the canonical form of a URI found in the From header field as they are for one found in the To header field or the P-Asserted-Identity header field.

First, implementations will ascertain if the user-portion of the URI constitutes a telephone number. 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 SIP URIs with the "user=phone" parameter conforms to the syntax of the TEL URI scheme (RFC 3966 [RFC3966]). It is also possible for a TEL URI to appear in SIP header fields outside the context of a SIP or SIPS URI (e.g., 'tel:+17005551008'). Thus, in

standards-compliant environments, numbers will be explicitly labeled by the use of TEL URIs or the 'user=phone' parameter.

Alternatively, implementations in environments that do not conform to those standards MAY follow local policies for identifying telephone numbers. For example, implementations could infer that the user part is a telephone number due to the presence of the '+' indicator at the start of the user-portion. Absent even that indication, if there are numbers present in the user-portion, implementations might conceivably 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. Implementations could also rely on external hints: for example, a verification service implementation could infer from the type of credential that signed a request that the signature must be over a telephone number.

Regardless of how the implementation detects telephone numbers, once a telephone number has been detected, implementations SHOULD follow the procedures in Section 8.3. If the URI field does not contain a telephone number, or if the result of the canonicalization of the From header field value does not form a valid E.164 telephone number, the authentication service and/or verification service SHOULD treat the entire URI as a SIP URI, and apply the procedures in Section 8.5. These URI normalization procedures are invoked to canonicalize the URI before it is included in a PASSport object in, for example, a "uri" claim. See Section 8.5 for that behavior.

## 8.2. Authority for Telephone Numbers

In order for telephone numbers to be used with the mechanism described in this document, authentication services must receive credentials from an authority for telephone numbers or telephone number ranges, and verification services must trust the authority employed by the authentication service that signs a request. Per Section 7.4, enrollment procedures and credential management are outside the scope of this document; approaches to credential management for telephone numbers are discussed in [I-D.ietf-stir-certificates].

## 8.3. Telephone Number Canonicalization Procedures

Once an implementation has identified a telephone number, it must construct a number string. That requires performing the following steps:

Implementations MUST drop any "+"s, any internal dashes, parentheses or other non-numeric characters, excepting only the "#" 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. This will be the case, for example, for nationally-specific service numbers (e.g. 911, 112); however, calls to those numbers are routed in a very strict fashion which ordinarily prevents them from reaching entities that don't understand the numbers.

Some domains may need to take unique steps to convert their numbers into a global format, and such transformations during canonicalization can also be made in accordance with specific policies used within a local domain. For example, one domain may only use local number formatting and need to convert all To/From header field user portions to E.164 by prepending country-code and region code digits; another domain might have prefixed usernames with trunk-routing codes, in which case the canonicalization will need to remove the prefix. This specification cannot anticipate all of the potential transformations that might be useful.

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*tn-char
tn-char = "#" / "*" / DIGIT
```

The resulting number string is used in the construction of the telephone number field(s) in a PASSporT object.

#### 8.4. Authority for Domain Names

To use a SIP URI as an identity in this mechanism requires authentication and verification systems to support standard

mechanisms for proving authority over a domain name: that is, the domain name in the host portion of the SIP URI.

A verifier MUST evaluate the correspondence between the user's identity and the signing credential by following the procedures defined in [RFC5922], Section 7.2. While [RFC5922] deals with the use of TLS and is specific to certificates, the procedures described are applicable to verifying identity if one substitutes the "hostname of the server" for the domain portion of the user's identity in the From header field of a SIP request with an Identity header field.

This process 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 (see [RFC5922] Section 7.8). Secondly, some SIP services may delegate SIP functions to a subordinate domain and utilize the procedures in [RFC3263] that allow requests for, say, 'example.com' to be routed to 'sip.example.com'. As a result, a user with the AoR 'sip:alice@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 address the second of these problems, a domain that deploys an authentication service on a subordinate host might 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:alice@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.

## 8.5. URI Normalization

Just as telephone numbers may undergo a number of syntactic transformations during transit, the same can happen to SIP and SIPS URIs without telephone numbers as they traverse certain intermediaries. Therefore, when generating a PASSporT object based on a SIP request, any SIP and SIPS URIs must be transformed into a canonical form which captures the address-of-record represented by the URI before they are provisioned in PASSporT claims such as "uri". The URI normalization procedures required are as follows.

Following the ABNF of RFC3261, the SIP or SIPS URI in question MUST discard all elements after the "hostport" of the URI, including all uri-parameters and escaped headers, from its syntax. Of the userinfo component of the SIP URI, only the user element will be retained: any password (and any leading ":" before the password) MUST be removed, and since this userinfo necessarily does not contain a telephone-subscriber component, no further parameters can appear in the user portion.

The hostport portion of the SIP or SIPS URI MUST similarly be stripped of any trailing port along with the ":" that proceeds the port, leaving only the host.

The ABNF of this canonical URI form (following the syntax defined in RFC3261) is:

```
    canon-uri = ( "sip" / "sips" ) ":" user "@" host
```

Finally, the URI will be subject to syntax-based URI normalization procedures of [RFC3986] Section 6.2.2. Implementations MUST perform case normalization (rendering the scheme, user, and host all lowercase) and percent-encoding normalization (decoding any percent-encoded octet that corresponds to an unreserved character, per [RFC3986] Section 2.3). However, note that normalization procedures face known challenges in some internationalized environments (see [I-D.ietf-iri-comparison]) and that perfect normalization of URIs may not be possible in those environments.

For future PASSporT applications, it may be desirable to provide an identifier without an attached protocol scheme. Future specifications that define PASSporT claims for SIP as a using protocol could use these basic procedures, but eliminate the scheme component. A more exact definition is left to future specifications.

## 9. Extensibility

As future requirements may warrant increasing the scope of the Identity mechanism, this specification specifies an optional "ppt" parameter of the Identity header field, which mirrors the "ppt" header in PASSporT. The "ppt" parameter value MUST consist of a token containing an extension specification, which denotes an extended set of one or more signed claims per the type extensibility mechanism specified in [I-D.ietf-stir-passport] Section 8. Note that per the guidance in that section, "ppt" is used only to enforce a mandatory extension: optional claims may be added to any PASSporT object without requiring the use of "ppt", but the compact form of PASSporT MUST NOT be used when optional claims are present in the PASSporT payload.

The potential for extensions is one the primary motivations for allowing the presence of multiple Identity header fields in the same SIP request. It is envisioned that future extensions might allow for alternate information to be signed, or to explicitly allow different parties to provide the signatures than the authorities envisioned by baseline STIR. A request might, for example, have one Identity added by an authentication service at the originating administrative domain, and then another Identity header field added by some further intermediary using a PASSporT extension. While this specification does not define any such specific purpose for multiple Identity header fields, implementations MUST support receiving multiple header fields for future compatibility reasons.

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 an extension; authors of extensions should provide appropriate extension-specific guidance to application developers on this point.

## 10. Backwards Compatibility with RFC4474

This specification introduces several significant changes from the RFC4474 version of the Identity header field. However, due to the problems enumerated in [I-D.rosenberg-sip-rfc4474-concerns], it is not believed that the original Identity header field has seen any deployment, or even implementation in deployed products.

As such, this mechanism contains no provisions for signatures generated with this specification to work with RFC4474-compliant implementations, nor any related backwards-compatibility provisions. Hypothetically, were an RFC4474-compliant implementation to receive messages containing this revised version of the Identity header

field, it would likely fail the request with a 436 response code due to the absence of an Identity-Info header field. Implementations of this specification, for debugging purposes, might interpret a 436 with a reason phrase of "Bad Identity-Info" as an indication that the request has failed because it reached a (hypothetical) RFC4474-compliant verification service.

## 11. Privacy Considerations

The purpose of this mechanism is to provide a reliable identification of the originator of a SIP request, specifically a cryptographic assurance that an authority asserts the originator can claim the URI the identity stipulated in the request. This URI may contain or imply 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], implementers 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 RFC3323 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 field is the same for private URIs as it is for any other sort of URIs. Similar practices could be used to support opportunistic signing of SIP requests for UA-integrated authentication services with self-signed certificates, though that is outside the scope of this specification and is left as a matter for future investigation.

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 value of the request; accordingly, using domains like 'anonymous.invalid' will not be usable by privacy services that simultaneously 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".

It is worth noting two features of this more anonymous form of identity. One can eliminate any identifying information in a domain through the use of the domain 'anonymous.invalid,' but we must then acknowledge that it is difficult for a domain to be both anonymous

and authenticated. The use of the "anonymous.invalid" domain entails that no corresponding authority for the domain can exist, and as a consequence, authentication service functions for that domain are meaningless. The second feature is more germane to the threats this document mitigates [RFC7375]. None of the relevant attacks, all of which rely on the attacker taking on the identity of a victim or hiding their identity using someone else's identity, are enabled by an anonymous identity. As such, the inability to assert an authority over an anonymous domain is irrelevant to our threat model.

[RFC3325] defines the "id" priv-value token, which is specific to the P-Asserted-Identity header field. The sort of assertion provided by the P-Asserted-Identity header field is very different from the Identity header field presented in this document. It contains additional information about the originator of a message that may go beyond what appears in the From header field; P-Asserted-Identity holds a definitive identity for the originator that is somehow known to a closed network of intermediaries. Presumably, that 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 field, and privacy services MUST NOT remove the Identity header field when a priv-value of "id" appears in a Privacy header field.

The full form of the PASSporT object provides the complete JSON objects used to generate the signed-identity-digest of the Identity header field value, including the canonicalized form of the telephone number of the originator of a call, if the signature is over a telephone number. In some contexts, local policy may require a canonicalization which differs substantially from the original From header field. Depending on those policies, potentially the full form of PASSporT 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 the object would need to be converted to the compact form 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 originator's identity could not be reassembled by a verifier, and thus the Identity signature validation process would fail, using P-Asserted-Identity with the full form of PASSporT in this fashion is NOT RECOMMENDED outside of environments where SIP requests will never leave the trust domain. As a side note, history shows that closed networks never stay closed and one should design their implementation assuming connectivity to the broader Internet.

Finally, note that unlike [RFC3325], the mechanism described in this specification adds no information to SIP requests that has privacy implications - apart from disclosing that an authentication service is willing to sign for an originator.

## 12. Security Considerations

This document describes a mechanism that provides a signature over the Date header field of SIP requests, parts of the To and From header fields, and when present any media keying material in the message body. In general, the considerations related to the security of these header fields are the same as those given in [RFC3261] for including header fields in tunneled 'message/sip' MIME bodies (see Section 23 of RFC3261 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. It addresses the solution-specific attacks against in-band solutions enumerated in [RFC7375] Section 4.1.

### 12.1. Protected Request Fields

The From header field value (in ordinary operations) indicates the identity of the originator of the message. 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. Note that in some deployments the identity of the originator may reside in P-Asserted-Id instead. The originator's identity is the key piece of information that this mechanism secures; the remainder of the signed parts of a SIP request are present to provide reference integrity and to prevent certain types of cut-and-paste attacks.

The Date header field value protects against cut-and-paste attacks, as described in [RFC3261], Section 23.4.2. That specification recommends that implementations notify the user of a potential security issue if the signed Date header field value is stale by an hour or more. To prevent cut-and-paste of recently-observed messages, this specification instead RECOMMENDS a shorter interval of sixty seconds. Implementations of this specification MUST NOT deem valid a request with an outdated Date header field. Note that per [RFC3893] Section 10 behavior, servers can keep state of recently received requests, and thus if an Identity header field is replayed by an attacker within the Date interval, verifiers can detect that it is spoofed because a message with an identical Date from the same source had recently been received.

It has been observed in the wild that some networks change the Date header field value of SIP requests in transit, and that alternative

behavior might be necessary to accommodate that use case. Verification services that observe a signature validation failure MAY therefore reconstruct the Date header field component of the signature from the "iat" carried in the full form of PASSporT: provided that time recorded by "iat" falls within the local policy for freshness that would ordinarily apply to the Date header, the verification service MAY treat the signature as valid, provided it keeps adequate state to detect recent replays. Note that this will require the inclusion of the full form of the PASSporT object by authentication services in networks where such failures are observed.

The To header field value provides the identity of the SIP user that this request originally targeted. Covering the identity in the To header field with the Identity signature serves two purposes. First, it prevents cut-and-paste attacks in which an Identity header field from a 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 To identity offers additional protection against cut-and-paste attacks beyond the Date header field. For example, without a signature over the To identity, an attacker who receives a call from a target could immediately cut-and-paste the Identity and From header field value from that INVITE into a new request to the target's voicemail service within the Date interval, and the voicemail service would have no way knowing that the Identity header field it received had been originally signed for a call intended for a different number. However, note the caveats below in Section 12.1.1.

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, launched with the REFER method or through social engineering, where the attacker receives a request from the target and reoriginates it to a third party. These 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 impersonation than is commonly used 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 media communication. This signature also reduces the potential for active eavesdropping attacks against the SIP media. In

environments where DTLS-SRTP is unsupported, however, no field is signed and no protections are provided.

#### 12.1.1. Protection of the To Header and Retargeting

Armed with the original value of the To header field, the recipient of a request may be tempted 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 direction 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 field for requests in the dialog only when the user is authorized to assert the identity given in the From header field, and if they are not, an Identity header field 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.

#### 12.2. Unprotected Request Fields

RFC4474 originally had protections for the Contact, Call-ID and CSeq. These are removed from RFC4474bis. The absence of these header field values creates some opportunities for determined attackers to impersonate based on cut-and-paste attacks; however, the absence of these header field values does not seem impactful to preventing the simple unauthorized claiming of an identity for the purposes of robocalling, voicemail hacking, or swatting, 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

field, an attacker could remove that Via header field 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 field 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.

### 12.3. Malicious Removal of Identity Headers

In the end analysis, the Identity header field cannot protect itself. Any attacker could remove the header field 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.

### 12.4. 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 parts of the header and body is provided by this mechanism only to prevent replay attacks. However, it is possible that applications relying on the presence of the Identity header field could leverage this integrity protection 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 field 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.

#### 12.5. Authorization and Transitional Strategies

Ultimately, the worth of an assurance provided by an Identity header field is limited by the security practices of the authentication service that issues the assurance. Relying on an Identity header field 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 field 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 header field 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 originator'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 source without an Identity header field with skepticism.

A verifier could consult some sort of directory that indicates 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 header fields 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 header fields are always good.

#### 12.6. 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, such as through the "ppt" name mechanism.

In the absence of signing the display-name, authentication services might check and validate it, and compare it to a list of acceptable display-names that may be used by the originator; 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.

### 13. IANA Considerations

This document contains a number of actions for IANA. Primarily, the previous references to RFC4474 in the sip-parameters registry should, unless specified otherwise below, be updated to point to [RFCthis].

#### 13.1. SIP Header Fields

The Identity-Info header in the SIP Header Fields registry should be marked as deprecated by [RFCThis].

Also, the Identity-Info header reserved the compact form "n" at its time of registration. Please remove that compact form from the registry. The Identity header however retains the compact form "y" reserved by RFC4474.

#### 13.2. SIP Response Codes

The Reason phrase for the 436 response default reason phrase should be changed from "Bad Identity-Info" to "Bad Identity Info" in the SIP Response Code registry.

The 437 "Unsupported Certificate" default reason phrase should be changed to "Unsupported Credential".

#### 13.3. Identity-Info Parameters

The IANA manages a registry for Identity-Info parameters. The specification asks the IANA to change the name of this registry to "Identity Parameters".

This specification defines one new value for the registry: "info" as defined in this specification in Section 7.3.

#### 13.4. Identity-Info Algorithm Parameter Values

This IANA manages an Identity-Info Algorithm Parameter Values registry which this specification deprecates. We request that the IANA deprecate and close this registry. Since the algorithms for signing PASSporTs are defined in [I-D.ietf-stir-passport] rather than in this specification, there is no longer a need for an algorithm parameter registry for the Identity header field.

#### 14. Acknowledgments

The authors would like to thank Adam Roach, Jim Schaad, Ning Zhang, Syed Ali, Olle Jacobson, Dave Frankel, Robert Sparks, Dave Crocker, 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, Eric Burger, Alan Ford, Christer Holmberg, Philippe Fouquart, Michael Hamer, Henning Schulzrinne, and Richard Barnes for their comments.

#### 15. Changes from RFC4474

The following are salient changes from the original RFC 4474:

Generalized the credential mechanism; credential enrollment, acquisition and trust 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; signing of key fingerprints in SDP is now included

Deprecated the Identity-Info header field and relocated its components into parameters of the Identity header field (which obsoletes the previous version of the header field)

The Identity header field can now appear multiple times in one request

Replaced previous signed-identity-digest format with PASSporT (signing algorithms now defined in a separate specification)

Revised status code descriptions

#### 16. References

## 16.1. Normative References

- [E.164] ITU-T, "The international public telecommunication numbering plan", E 164, February 2005, <<https://www.itu.int/rec/T-REC-E.164/en>>.
- [I-D.ietf-stir-passport]  
Wendt, C. and J. Peterson, "Personal Assertion Token (PASSporT)", draft-ietf-stir-passport-10 (work in progress), October 2016.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<http://www.rfc-editor.org/info/rfc2119>>.
- [RFC3261] Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., Handley, M., and E. Schooler, "SIP: Session Initiation Protocol", RFC 3261, DOI 10.17487/RFC3261, June 2002, <<http://www.rfc-editor.org/info/rfc3261>>.
- [RFC3263] Rosenberg, J. and H. Schulzrinne, "Session Initiation Protocol (SIP): Locating SIP Servers", RFC 3263, DOI 10.17487/RFC3263, June 2002, <<http://www.rfc-editor.org/info/rfc3263>>.
- [RFC3966] Schulzrinne, H., "The tel URI for Telephone Numbers", RFC 3966, DOI 10.17487/RFC3966, December 2004, <<http://www.rfc-editor.org/info/rfc3966>>.
- [RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, RFC 3986, DOI 10.17487/RFC3986, January 2005, <<http://www.rfc-editor.org/info/rfc3986>>.
- [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, DOI 10.17487/RFC5280, May 2008, <<http://www.rfc-editor.org/info/rfc5280>>.
- [RFC5922] Gurbani, V., Lawrence, S., and A. Jeffrey, "Domain Certificates in the Session Initiation Protocol (SIP)", RFC 5922, DOI 10.17487/RFC5922, June 2010, <<http://www.rfc-editor.org/info/rfc5922>>.

## 16.2. Informative References

- [I-D.ietf-iri-comparison]  
Masinter, L. and M. D&#258;&#378;rst, "Comparison, Equivalence and Canonicalization of Internationalized Resource Identifiers", draft-ietf-iri-comparison-02 (work in progress), October 2012.
- [I-D.ietf-stir-certificates]  
Peterson, J. and S. Turner, "Secure Telephone Identity Credentials: Certificates", draft-ietf-stir-certificates-11 (work in progress), October 2016.
- [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, DOI 10.17487/RFC2585, May 1999, <<http://www.rfc-editor.org/info/rfc2585>>.
- [RFC3323] Peterson, J., "A Privacy Mechanism for the Session Initiation Protocol (SIP)", RFC 3323, DOI 10.17487/RFC3323, November 2002, <<http://www.rfc-editor.org/info/rfc3323>>.
- [RFC3325] Jennings, C., Peterson, J., and M. Watson, "Private Extensions to the Session Initiation Protocol (SIP) for Asserted Identity within Trusted Networks", RFC 3325, DOI 10.17487/RFC3325, November 2002, <<http://www.rfc-editor.org/info/rfc3325>>.
- [RFC3893] Peterson, J., "Session Initiation Protocol (SIP) Authenticated Identity Body (AIB) Format", RFC 3893, DOI 10.17487/RFC3893, September 2004, <<http://www.rfc-editor.org/info/rfc3893>>.

- [RFC4474] Peterson, J. and C. Jennings, "Enhancements for Authenticated Identity Management in the Session Initiation Protocol (SIP)", RFC 4474, DOI 10.17487/RFC4474, August 2006, <<http://www.rfc-editor.org/info/rfc4474>>.
- [RFC4501] Josefsson, S., "Domain Name System Uniform Resource Identifiers", RFC 4501, DOI 10.17487/RFC4501, May 2006, <<http://www.rfc-editor.org/info/rfc4501>>.
- [RFC4916] Elwell, J., "Connected Identity in the Session Initiation Protocol (SIP)", RFC 4916, DOI 10.17487/RFC4916, June 2007, <<http://www.rfc-editor.org/info/rfc4916>>.
- [RFC5234] Crocker, D., Ed. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, DOI 10.17487/RFC5234, January 2008, <<http://www.rfc-editor.org/info/rfc5234>>.
- [RFC5393] Sparks, R., Ed., Lawrence, S., Hawrylyshen, A., and B. Campen, "Addressing an Amplification Vulnerability in Session Initiation Protocol (SIP) Forking Proxies", RFC 5393, DOI 10.17487/RFC5393, December 2008, <<http://www.rfc-editor.org/info/rfc5393>>.
- [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, DOI 10.17487/RFC5763, May 2010, <<http://www.rfc-editor.org/info/rfc5763>>.
- [RFC6698] Hoffman, P. and J. Schlyter, "The DNS-Based Authentication of Named Entities (DANE) Transport Layer Security (TLS) Protocol: TLSA", RFC 6698, DOI 10.17487/RFC6698, August 2012, <<http://www.rfc-editor.org/info/rfc6698>>.
- [RFC6973] Cooper, A., Tschofenig, H., Aboba, B., Peterson, J., Morris, J., Hansen, M., and R. Smith, "Privacy Considerations for Internet Protocols", RFC 6973, DOI 10.17487/RFC6973, July 2013, <<http://www.rfc-editor.org/info/rfc6973>>.
- [RFC7159] Bray, T., Ed., "The JavaScript Object Notation (JSON) Data Interchange Format", RFC 7159, DOI 10.17487/RFC7159, March 2014, <<http://www.rfc-editor.org/info/rfc7159>>.

- [RFC7258] Farrell, S. and H. Tschofenig, "Pervasive Monitoring Is an Attack", BCP 188, RFC 7258, DOI 10.17487/RFC7258, May 2014, <<http://www.rfc-editor.org/info/rfc7258>>.
- [RFC7340] Peterson, J., Schulzrinne, H., and H. Tschofenig, "Secure Telephone Identity Problem Statement and Requirements", RFC 7340, DOI 10.17487/RFC7340, September 2014, <<http://www.rfc-editor.org/info/rfc7340>>.
- [RFC7375] Peterson, J., "Secure Telephone Identity Threat Model", RFC 7375, DOI 10.17487/RFC7375, October 2014, <<http://www.rfc-editor.org/info/rfc7375>>.
- [RFC7515] Jones, M., Bradley, J., and N. Sakimura, "JSON Web Signature (JWS)", RFC 7515, DOI 10.17487/RFC7515, May 2015, <<http://www.rfc-editor.org/info/rfc7515>>.
- [RFC7519] Jones, M., Bradley, J., and N. Sakimura, "JSON Web Token (JWT)", RFC 7519, DOI 10.17487/RFC7519, May 2015, <<http://www.rfc-editor.org/info/rfc7519>>.

## Authors' Addresses

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

Email: [jon.peterson@neustar.biz](mailto:jon.peterson@neustar.biz)

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

Email: [fluffy@cisco.com](mailto:fluffy@cisco.com)

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

Email: [ekr@rtfm.com](mailto:ekr@rtfm.com)

Chris Wendt  
Comcast  
One Comcast Center  
Philadelphia, PA 19103  
USA

Email: [chris-ietf@chriswendt.net](mailto:chris-ietf@chriswendt.net)