

Network Working Group
Internet-Draft
Intended status: Informational
Expires: May 7, 2020

E. Rescorla
Mozilla
J. Peterson
Neustar
November 4, 2019

STIR Out-of-Band Architecture and Use Cases draft-ietf-stir-oob-06

Abstract

The PASSport format defines a token that can be carried by signaling protocols, including SIP, to cryptographically attest the identity of callers. Not all telephone calls use Internet signaling protocols, however, and some calls use them for only part of their signaling path, or cannot reliably deliver SIP header fields end-to-end. This document describes use cases that require the delivery of PASSport objects outside of the signaling path, and defines architectures and semantics to provide this functionality.

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 <https://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 May 7, 2020.

Copyright Notice

Copyright (c) 2019 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 (<https://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	4
3.	Operating Environments	4
4.	Dataflows	5
5.	Use Cases	6
5.1.	Case 1: VoIP to PSTN Call	7
5.2.	Case 2: Two Smart PSTN endpoints	7
5.3.	Case 3: PSTN to VoIP Call	7
5.4.	Case 4: Gateway Out-of-band	8
5.5.	Case 5: Enterprise Call Center	9
6.	Storing and Retrieving PASSporTs	9
6.1.	Storage	10
6.2.	Retrieval	11
7.	Solution Architecture	12
7.1.	Credentials and Phone Numbers	12
7.2.	Call Flow	12
7.3.	Security Analysis	13
7.4.	Substitution Attacks	14
7.5.	Rate Control for CPS Storage	15
8.	Authentication and Verification Service Behavior for Out-of-Band	16
8.1.	Authentication Service	16
8.2.	Verification Service	18
8.3.	Gateway Placement Services	19
9.	Example HTTPS Interface to the CPS	19
10.	CPS Discovery	21
11.	Encryption Key Lookup	22
12.	Acknowledgments	24
13.	IANA Considerations	24
14.	Security Considerations	24
15.	Informative References	24
	Authors' Addresses	26

[1.](#) Introduction

The STIR problem statement [[RFC7340](#)] describes widespread problems enabled by impersonation in the telephone network, including illegal robocalling, voicemail hacking, and swatting. As telephone services are increasingly migrating onto the Internet, and using Voice over IP (VoIP) protocols such as SIP [[RFC3261](#)], it is necessary for these protocols to support stronger identity mechanisms to prevent

impersonation. For example, [[RFC8224](#)] defines a SIP Identity header field capable of carrying PASSporT [[RFC8225](#)] objects in SIP as a means to cryptographically attest that the originator of a telephone call is authorized to use the calling party number (or, for native SIP cases, SIP URI) associated with the originator of the call.

Not all telephone calls use SIP today, however, and even those that do use SIP do not always carry SIP signaling end-to-end. Calls from telephone numbers still routinely traverse the Public Switched Telephone Network (PSTN) at some point. Broadly, calls fall into one of three categories:

1. One or both of the endpoints is actually a PSTN endpoint.
2. Both of the endpoints are non-PSTN (SIP, Jingle, ...) but the call transits the PSTN at some point.
3. Non-PSTN calls which do not transit the PSTN at all (such as native SIP end-to-end calls).

The first two categories represent the majority of telephone calls associated with problems like illegal robocalling: many robocalls today originate on the Internet but terminate at PSTN endpoints. However, the core network elements that operate the PSTN are legacy devices that are unlikely to be upgradable at this point to support an in-band authentication system. As such, those devices largely cannot be modified to pass signatures originating on the Internet--or indeed any inband signaling data--intact. Even if fields for tunneling arbitrary data can be found in traditional PSTN signaling, in some cases legacy elements would strip the signatures from those fields; in others, they might damage them to the point where they cannot be verified. For those first two categories above, any in-band authentication scheme does not seem practical in the current environment.

But while the core network of the PSTN remains fixed, the endpoints of the telephone network are becoming increasingly programmable and sophisticated. Landline "plain old telephone service" deployments, especially in the developed world, are shrinking, and increasingly being replaced by three classes of intelligent devices: smart phones, IP PBXs, and terminal adapters. All three are general purpose computers, and typically all three have Internet access as well as access to the PSTN; they may be used for residential, mobile, or enterprise telephone services. Additionally, various kinds of gateways increasingly front for deployments of legacy PBX and PSTN switches. All of this provides a potential avenue for building an authentication system that implements stronger identity while leaving PSTN systems intact.

This capability also provides an ideal transitional technology while in-band STIR adoption is ramping up. It permits early adopters to use the technology even when intervening network elements are not yet STIR-aware, and through various kinds of gateways, it may allow providers with a significant PSTN investment to still secure their calls with STIR.

Then techniques described in this document therefore build on the PASSporT [[RFC8225](#)] mechanism and the work of [[RFC8224](#)] to describe a way that a PASSporT object created in the originating network of a call can reach the terminating network even when it cannot be carried end-to-end in-band in the call signaling. This relies on a new service defined in this document called a Call Placement Service (CPS) that permits the PASSporT object to be stored during call processing and retrieved for verification purposes.

Potential implementors should note that this document merely defines the operating environments in which this out-of-band STIR mechanism is intended to operate. It provides use cases, gives a broad description of the components and a potential solution architecture. Various environments may have their own security requirements: a public deployment of out-of-band STIR faces far greater challenges than a constrained intranetwork deployment. To flesh out the storage and retrieval of PASSporTs in the CPS within this context, this document includes a strawman protocol suitable for that purpose. Deploying this framework in any given environment would require additional specification outside the scope of the current document.

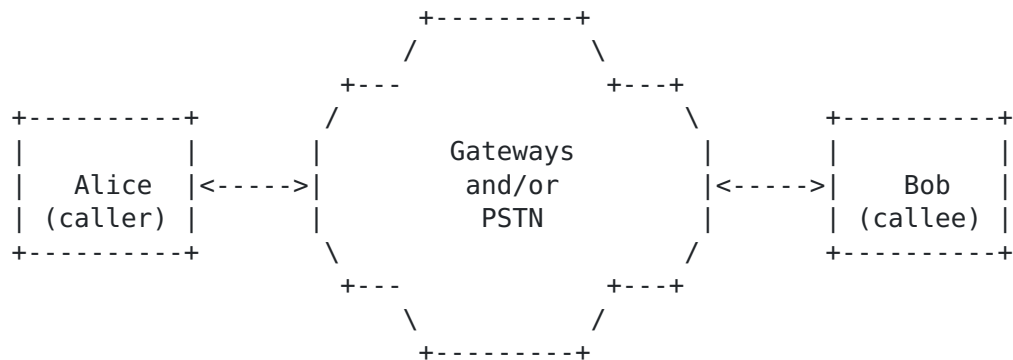
2. Terminology

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.

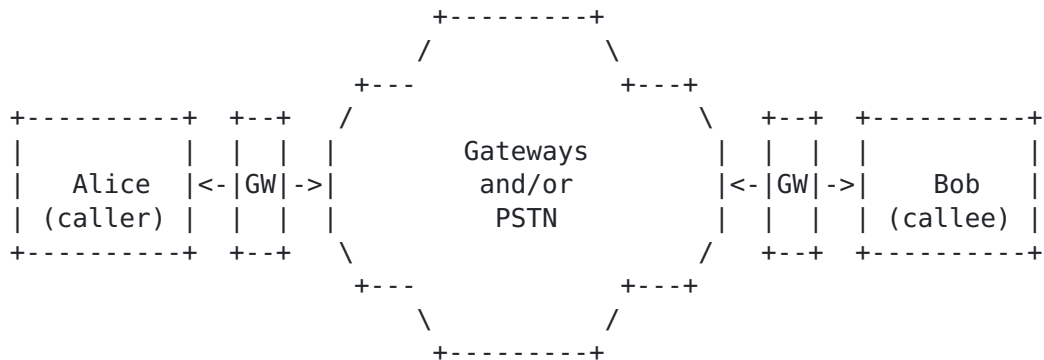
3. Operating Environments

This section describes the environments in which the proposed out-of-band STIR mechanism is intended to operate. In the simplest setting, Alice is calling Bob, and her call is routed through some set of gateways and/or the PSTN which do not support end-to-end delivery of STIR. Both Alice and Bob have smart devices which can access the Internet (perhaps enterprise devices, or even end user ones), but they do not have a clear telephone signaling connection between them: Alice cannot inject any data into signaling which Bob can read, with the exception of the asserted destination and origination E.164 numbers. The calling party number might originate from her own

device or from the network. These numbers are effectively the only data that can be used for coordination between the endpoints.



In a more complicated setting, Alice and/or Bob may not have a smart or programmable device, but instead just a traditional telephone. However, one or both of them are behind a STIR-aware gateway that can participate in out-of-band coordination, as shown below:



In such a case, Alice might have an analog (e.g., PSTN) connection to her gateway/ switch which is responsible for her identity. Similarly, the gateway would verify Alice's identity, generate the right calling party number information and provide that number to Bob using ordinary Plain Ol' Telephone Service (POTS) mechanisms.

4. Dataflows

Because in these operating environments endpoints cannot pass cryptographic information to one another directly through signaling, any solution must involve some rendezvous mechanism to allow endpoints to communicate. We call this rendezvous service a "call placement service" (CPS), a service where a record of call placement, in this case a PASSport, can be stored for future retrieval. In principle this service could communicate any information, but minimally we expect it to include a full-form PASSport that attests

the caller, callee, and the time of the call. The callee can use the existence of a PASSporT for a given incoming call as rough validation of the asserted origin of that call. (See [Section 11](#) for limitations of this design.)

This architecture does not mandate that any particular sort of entity operate a CPS, or mandate any means to discover a CPS. A CPS could be run internally within a network, or made publicly available. One or more CPSes could be run by a carrier, as repositories for PASSporTs for calls sent to its customers, or a CPS could be built-in to an enterprise PBX, or even a smartphone. To the degree possible, it is here specified generically, as an idea that may have applicability to a variety of STIR deployments.

There are roughly two plausible dataflow architectures for the CPS:

1. The callee registers with the CPS. When the caller wishes to place a call to the callee, it sends the PASSporT to the CPS, which immediately forwards it to the callee, or,
2. The caller stores the PASSporT with the CPS at the time of call placement. When the callee receives the call, it contacts the CPS and retrieves the PASSporT.

While the first architecture is roughly isomorphic to current VoIP protocols, it shares their drawbacks. Specifically, the callee must maintain a full-time connection to the CPS to serve as a notification channel. This comes with the usual networking costs to the callee and is especially problematic for mobile endpoints. Indeed, if the endpoints had the capabilities to implement such an architecture, they could surely just use SIP or some other protocol to set up a secure session; even if the media were going through the traditional PSTN, a "shadow" SIP session could convey the PASSporT. Thus, we focus on the second architecture in which the PSTN incoming call serves as the notification channel and the callee can then contact the CPS to retrieve the PASSporT. In specialized environments, for example a call center that receives a large volume of incoming calls that originated in the PSTN, the notification channel approach might be viable.

5. Use Cases

The following are the motivating use cases for this mechanism. Bear in mind that just as in [\[RFC8224\]](#) there may be multiple Identity headers in a single SIP INVITE, so there may be multiple PASSporTs in this out-of-band mechanism associated with a single call. For example, a SIP user agent might create a PASSporT for a call with an end user credential, and as the call exits the originating

administrative domain the network authentication service might create its own PASSporT for the same call. As such, these use cases may overlap in the processing of a single call.

5.1. Case 1: VoIP to PSTN Call

A call originates in a SIP environment in a STIR-aware administrative domain. The local authentication service for that administrative domain creates a PASSporT which is carried in band in the call per [\[RFC8224\]](#). The call is routed out of the originating administrative domain and reaches a gateway to the PSTN. Eventually, the call will terminate on a mobile smartphone that supports this out-of-band mechanism.

In this use case, the originating authentication service can store the PASSporT with the appropriate CPS for the target telephone number as a fallback in case SIP signaling will not reach end-to-end. When the destination mobile smartphone receives the call over the PSTN, it consults the CPS and discovers a PASSporT from the originating telephone number waiting for it. It uses this PASSporT to verify the calling party number.

5.2. Case 2: Two Smart PSTN endpoints

A call originates with an enterprise PBX that has both Internet access and a built-in gateway to the PSTN, which communicates through traditional telephone signaling protocols. The PBX immediately routes the call to the PSTN, but before it does, it provisions a PASSporT on the CPS associated with the target telephone number.

After normal PSTN routing, the call lands on a smart mobile handset that supports the STIR out-of-band mechanism. It queries the appropriate CPS over the Internet to determine if a call has been placed to it by a STIR-aware device. It finds the PASSporT provisioned by the enterprise PBX and uses it to verify the calling party number.

5.3. Case 3: PSTN to VoIP Call

A call originates with an enterprise PBX that has both Internet access and a built-in gateway to the PSTN. It will immediately route the call to the PSTN, but before it does, it provisions a PASSporT with the CPS associated with the target telephone number. However, it turns out that the call will eventually route through the PSTN to an Internet gateway, which will translate this into a SIP call and deliver it to an administrative domain with a STIR verification service.

In this case, there are two subcases for how the PASSport might be retrieved. In subcase 1, the Internet gateway that receives the call from the PSTN could query the appropriate CPS to determine if the original caller created and provisioned a PASSport for this call. If so, it can retrieve the PASSport and, when it creates a SIP INVITE for this call, add a corresponding Identity header field per [\[RFC8224\]](#). When the SIP INVITE reaches the destination administrative domain, it will be able to verify the PASSport normally. Note that to avoid discrepancies with the Date header field value, only full-form PASSport should be used for this purpose. In subcase 2, the gateway does not retrieve the PASSport itself, but instead the verification service at the destination administrative domain does so. Subcase 1 would perhaps be valuable for deployments where the destination administrative domain supports in-band STIR but not out-of-band STIR.

5.4. Case 4: Gateway Out-of-band

A call originates in the SIP world in a STIR-aware administrative domain. The local authentication service for that administrative domain creates a PASSport which is carried in band in the call per [\[RFC8224\]](#). The call is routed out of the originating administrative domain and eventually reaches a gateway to the PSTN.

In this case, the originating authentication service does not support the out-of-band mechanism, so instead the gateway to the PSTN extracts the PASSport from the SIP request and provisions it to the CPS. (When the call reaches the gateway to the PSTN, the gateway might first check the CPS to see if a PASSport object had already been provisioned for this call, and only provision a PASSport if none is present).

Ultimately, the call may terminate on the PSTN, or be routed back to a SIP environment. In the former case, perhaps the destination endpoint queries the CPS to retrieve the PASSport provisioned by the first gateway. Or if the call ultimately returns to a SIP environment, it might be the gateway from the PSTN back to the Internet that retrieves the PASSport from the CPS and attaches it to the new SIP INVITE it creates, or it might be the terminating administrative domain's verification service that checks the CPS when an INVITE arrives with no Identity header field. Either way the PASSport can survive the gap in SIP coverage caused by the PSTN leg of the call.

5.5. Case 5: Enterprise Call Center

A call originates from a mobile user, and a STIR authentication service operated by their carrier creates a PASSporT for the call. As the carrier forwards the call via SIP, it attaches the PASSporT to the SIP call with an Identity header field. As a fallback in case the call will not go end-to-end over SIP, the carrier also stores the PASSporT in a CPS.

The call is then routed over SIP for a time, before it transitions to the PSTN and ultimately is handled by a legacy PBX at a high-volume call center. The call center supports the out-of-band service, and has a high-volume interface to a CPS to retrieve PASSporTs for incoming calls; agents at the call center use a general purpose computer to manage inbound calls and can receive STIR notifications through it. When the PASSporT arrives at the CPS, it is sent through a subscription/notification interface to a system that can correlate incoming calls with valid PASSporTs. The call center agent sees that a valid call from the originating number has arrived.

6. Storing and Retrieving PASSporTs

The use cases show a variety of entities accessing the CPS to store and retrieve PASSporTs. The question of how the CPS authorizes the storage and retrieval of PASSporT is thus a key design decision in the architecture. The STIR architecture assumes that service providers and in some cases end user devices will have credentials suitable for attesting authority over telephone numbers per [\[RFC8226\]](#). These credentials provide the most obvious way that a CPS can authorize the storage and retrieval of PASSporTs. However, as use cases 3, 4 and 5 in [Section 5](#) show, it may sometimes make sense for the entity storing or retrieving PASSporTs to be an intermediary rather than a device associated with either the originating or terminating side of a call, and those intermediaries often would not have access to STIR credentials covering the telephone numbers in question. Requiring authorization based on a credential to store PASSporTs is therefore undesirable, though potentially acceptable if sufficient steps are taken to mitigate any privacy risk of leaking data.

It is an explicit design goal of this mechanism to minimize the potential privacy exposure of using a CPS. Ideally, the out-of-band mechanism should not result in a worse privacy situation than in-band [\[RFC8224\]](#) STIR: for in-band, we might say that a SIP entity is authorized to receive a PASSporT if it is an intermediate or final target of the routing of a SIP request. As the originator of a call cannot necessarily predict the routing path a call will follow, an

out-of-band mechanism could conceivably even improve on the privacy story.

Broadly, the architecture recommended here thus is one focused on permitting any entity to store encrypted PASSporTs at the CPS, indexed under the called number. PASSporTs will be encrypted with an encryption key signed by the public key associated with the called number, so these PASSporTs may safely be retrieved by any entity, as only holders of the corresponding private key will be able to decrypt the PASSporT. This also prevents the CPS itself from learning the contents of PASSporTs, and thus metadata about calls in progress, which makes the CPS a less attractive target for pervasive monitoring (see [\[RFC7258\]](#)). As a first step, transport-level security can provide confidentiality from eavesdroppers for both the storage and retrieval of PASSporTs. To bolster the privacy story, prevent denial-of-service flooding of the CPS, and to complicate traffic analysis, a few additional mechanisms are also recommended below.

6.1. Storage

There are a few dimensions to authorizing the storage of PASSporTs. Encrypting PASSporTs prior to storage entails that a CPS has no way to tell if a PASSporT is valid; it simply conveys encrypted blocks that it cannot access itself, and can make no authorization decision based on the PASSporT contents. There is certainly no prospect for the CPS to verify the PASSporTs itself.

Note that this architecture requires clients that store PASSporTs to have access to an encryption key associated with the intended called party to be used to encrypt the PASSporT. Discovering this key requires the existence of a key lookup service (see [Section 11](#)); depending on how the CPS is architected, however, some kind of key store or repository could be implemented adjacent to it, and perhaps even incorporated into its operation. Key discovery is made more complicated by the fact that there can potentially be multiple entities that have authority over a telephone number: a carrier, a reseller, an enterprise, and an end user might all have credentials permitting them to attest that they are allowed to originate calls from a number, say. PASSporTs for out-of-band use therefore might need to be encrypted with multiple keys in the hopes that one will be decipherable by the relying party.

Again, the most obvious way to authorize storage is to require the originator to authenticate themselves to the CPS with their STIR credential. However, since the call is indexed at the CPS under the called number, this can weaken the privacy story of the architecture, as it reveals to the CPS both the identity of the caller and the callee. Moreover, it does not work for the gateway use cases

described above; to support those use cases, we must effectively allow any entity to store PASSporTs at a CPS. This does not degrade the anti-impersonation security of STIR, because entities who do not possess the necessary credentials to sign the PASSporT will not be able to create PASSporTs that will be treated as valid by verifiers. In this architecture, it does not matter whether the CPS received a PASSporT from the authentication service that created it or from an intermediary gateway downstream in the routing path as in case 4 above. However, if literally anyone can store PASSporTs in the CPS, an attacker could easily flood the CPS with millions of bogus PASSporTs indexed under a calling number, and thereby prevent the called party from finding a valid PASSporT for an incoming call buried in a haystack of fake entries.

The solution architecture must therefore include some sort of traffic control system to prevent flooding. Preferably, this should not require authenticating the source, as this will reveal to the CPS both the source and destination of traffic. A potential solution is discussed below in [Section 7.5](#).

6.2. Retrieval

For retrieval of PASSporTs, this architecture assumes that clients will contact the CPS through some sort of polling or notification interface to receive all current PASSporTs for calls destined to a particular telephone number, or block of numbers.

As PASSporTs stored at the CPS are encrypted with a key belonging to the intended destination, the CPS can safely allow anyone to download PASSporTs for a called number without much fear of compromising private information about calls in progress - provided that the CPS always returns at least one encrypted blob in response to a request, even if there was no call in progress. Otherwise, entities could poll the CPS constantly, or eavesdrop on traffic, to learn whether or not calls were in progress. The CPS MUST generate at least one unique and plausible encrypted response to all retrieval requests, and these dummy encrypted PASSporTs MUST NOT be repeated for later calls.

Because the entity placing a call may discover multiple keys associated with the called party number, multiple valid PASSporTs may be stored in the CPS. A particular called party who retrieves PASSporTs from the CPS may have access to only one of those keys. Thus, the presence of one or more PASSporTs that the called party cannot decrypt - which would be indistinguishable from the "dummy" PASSporTs created by the CPS when no calls are in progress - does not entail that there is no call in progress. A retriever likely will

need to decrypt all PASSporTs retrieved from the CPS, and may find only one that is valid.

Note that in out-of-band call forwarding cases, special behavior is required to manage the relationship between PASSporTs using the diversion extension [[I-D.ietf-stir-passport-divert](#)]. The originating authentication service would encrypt the initial PASSporT with the public encryption key of the intended destination, but once a call is forwarded, it may go to a destination that does not possess the corresponding private key and thus could not decrypt the original PASSporT. This requires the retargeting entity to generate encrypted PASSporTs that show a secure chain of diversion: a retargeting storer SHOULD use the "div-o" PASSporT type, with its "opt" extension, as specified in [[I-D.ietf-stir-passport-divert](#)] in order to nest the original PASSporT within the encrypted diversion PASSporT.

[7.](#) Solution Architecture

In this section, we discuss a high-level architecture for providing the service described in the previous sections. This discussion is deliberately sketchy, focusing on broad concepts and skipping over details. The intent here is merely to provide an overall architecture, not an implementable specification. A more concrete example of how this might be specified is given in [Section 9](#).

[7.1.](#) Credentials and Phone Numbers

We start from the premise of the STIR problem statement [[RFC7340](#)] that phone numbers can be associated with credentials which can be used to attest ownership of numbers. For purposes of exposition, we will assume that ownership is associated with the endpoint (e.g., a smartphone) but it might well be associated with a provider or gateway acting for the endpoint instead. It might be the case that multiple entities are able to act for a given number, provided that they have the appropriate authority. [[RFC8226](#)] describes a credential system suitable for this purpose; the question of how an entity is determined to have control of a given number is out of scope for the current document.

[7.2.](#) Call Flow

An overview of the basic calling and verification process is shown below. In this diagram, we assume that Alice has the number +1.111.555.1111 and Bob has the number +2.222.555.2222.


```
Alice                      Call Placement Service                      Bob
-----
Store PASSporT for 2.222.555.2222 -->

Call from 1.111.555.1111 ----->

                                <----- Request PASSporT(s)
                                for 2.222.555.2222

                                Obtain Encrypted PASSporT ----->
                                (2.222.555.2222, 1.111.555.1111)

                                [Ring phone with callerid
                                = 1.111.555.1111]
```

When Alice wishes to make a call to Bob, she contacts the CPS and stores an encrypted PASSporT on the CPS indexed under Bob's number. The CPS then awaits retrievals for that number.

When Alice places the call, Bob's phone would usually ring and display Alice's number (+1.111.555.1111), which is informed by the existing PSTN mechanisms for relaying a calling party number (e.g., the CIN field of the IAM). Instead, Bob's phone transparently contacts the CPS and requests any current PASSporTs for calls to his number. The CPS responds with any such PASSporTs (assuming they exist). If such a PASSporT exists, and the verification service in Bob's phone decrypts it using his private key, validates it, then Bob's phone can present the calling party number information as valid. Otherwise, the call is unverifiable. Note that this does not necessarily mean that the call is bogus; because we expect incremental deployment, many legitimate calls will be unverifiable.

7.3. Security Analysis

The primary attack we seek to prevent is an attacker convincing the callee that a given call is from some other caller C. There are two scenarios to be concerned with:

1. The attacker wishes to impersonate a target when no call from that target is in progress.
2. The attacker wishes to substitute himself for an existing call setup as described in [Section 7.4](#).

If an attacker can inject fake PASSporT into the CPS or in the communication from the CPS to the callee, he can mount either attack.

As PASSporTs should be digitally signed by an appropriate authority for the number and verified by the callee (see [Section 7.1](#)), this should not arise in ordinary operations. For privacy and robustness reasons, using TLS [\[RFC8446\]](#) on the originating side when storing the PASSporT at the CPS is RECOMMENDED.

The entire system depends on the security of the credential infrastructure. If the authentication credentials for a given number are compromised, then an attacker can impersonate calls from that number. However, that is no different from in-band [\[RFC8224\]](#) STIR.

A secondary attack we must also prevent is denial-of-service against the CPS, which requires some form of rate control solution that will not degrade the privacy properties of the architecture.

[7.4.](#) Substitution Attacks

All that receipt of the PASSporT from the CPS proves to the called party is that Alice is trying to call Bob (or at least was as of very recently) - it does not prove that any particular incoming call is from Alice. Consider the scenario in which we have a service which provides an automatic callback to a user-provided number. In that case, the attacker can try to arrange for a false caller-id value, as shown below:

Attacker	Callback Service	CPS	Bob

Place call to Bob ----->			
	Store PASSporT for CS:Bob ----->		
Call from CS (forged caller-id info) ----->			
	Call from CS -----> X		
		<-- Retrieve PASSporT for CS:Bob	
	PASSporT for CS:Bob ----->		
		[Ring phone with callerid = 111.555.1111]	

In order to mount this attack, the attacker contacts the Callback Service (CS) and provides it with Bob's number. This causes the CS to initiate a call to Bob. As before, the CS contacts the CPS to

insert an appropriate PASSporT and then initiates a call to Bob. Because it is a valid CS injecting the PASSporT, none of the security checks mentioned above help. However, the attacker simultaneously initiates a call to Bob using forged caller-id information corresponding to the CS. If he wins the race with the CS, then Bob's phone will attempt to verify the attacker's call (and succeed since they are indistinguishable) and the CS's call will go to busy/voice mail/call waiting.

In order to prevent a passive attacker from using traffic analysis or similar means to learn precisely when a call is placed, it is essential that the connection between the caller and the CPS be encrypted as recommended above. Callers could store dummy PASSporTs at the CPS at random intervals in order to make it more difficult for an eavesdropper to use traffic analysis to determine that a call was about to be placed.

Note that in a SIP environment, the callee might notice that there were multiple INVITEs and thus detect this attack, but in some PSTN interworking scenarios, or highly intermediated networks, only one call setup attempt will reach the target. Also note that the success of this substitution attack depends on the attacker landing their call within the narrow window that the PASSporT is retained in the CPS, so shortening that window will reduce the opportunity for the attack. Finally, smart endpoints could implement some sort of state coordination to ensure that both sides believe the call is in progress, though methods of supporting that are outside the scope of this document.

7.5. Rate Control for CPS Storage

In order to prevent the flooding of a CPS with bogus PASSporTs, we propose the use of "blind signatures" (see [[RFC5636](#)]). A sender will initially authenticate to the CPS using its STIR credentials, and acquire a signed token from the CPS that will be presented later when storing a PASSporT. The flow looks as follows:

Sender	CPS
Authenticate to CPS ----->	
Blinded(K_temp) ----->	
<----- Sign(K_cps, Blinded(K_temp))	
[Disconnect]	
Sign(K_cps, K_temp)	
Sign(K_temp, E(K_receiver, PASSporT)) --->	

At an initial time when no call is yet in progress, a potential client connects to the CPS, authenticates, and sends a blinded version of a freshly generated public key. The CPS returns a signed version of that blinded key. The sender can then unblind the key and gets a signature on K_temp from the CPS.

Then later, when a client wants to store a PASSporT, it connects to the CPS anonymously (preferably over a network connection that cannot be correlated with the token acquisition) and sends both the signed K_temp and its own signature over the encrypted PASSporT. The CPS verifies both signatures and if they verify, stores the encrypted passport (discarding the signatures).

This design lets the CPS rate limit how many PASSporTs a given sender can store just by counting how many times K_temp appears; perhaps CPS policy might reject storage attempts and require acquisition of a new K_temp after storing more than a certain number of PASSporTs indexed under the same destination number in a short interval. This does not of course allow the CPS to tell when bogus data is being provisioned by an attacker, simply the rate at which data is being provisioned. Potentially, feedback mechanisms could be developed that would allow the called parties to tell the CPS when they are receiving unusual or bogus PASSporTs.

This architecture also assumes that the CPS will age out PASSporTs. A CPS SHOULD NOT keep any stored PASSporT for no longer than a value that might be selected for the verification service policy for freshness of the "iat" value as described in [\[RFC8226\]](#). Any reduction in this window makes substitution attacks (see [Section 7.4](#)) harder to mount, but making the window too small might conceivably age PASSporTs out while a heavily redirected call is still alerting.

8. Authentication and Verification Service Behavior for Out-of-Band

[RFC8224] defines an authentication service and a verification service as functions that act in the context of SIP requests and responses. This specification thus provides a more generic description of authentication service and verification service behavior that might or might not involve any SIP transactions, but depends only on placing a request for communications from an originating identity to one or more destination identities.

8.1. Authentication Service

Out-of-band authentication services perform steps similar to those defined in [\[RFC8224\]](#) with some exceptions:

Step 1: The authentication service MUST determine whether it is authoritative for the identity of the originator of the request, that is, the identity it will populate in the "orig" claim of the PASSporT. It can do so only if it possesses the private key of one or more credentials that can be used to sign for that identity, be it a domain or a telephone number or some other identifier. For example, the authentication service could hold the private key associated with a STIR certificate [[RFC8225](#)].

Step 2: The authentication service MUST determine that the originator of communications can claim the originating identity. This is a policy decision made by the authentication service that depends on its relationship to the originator. For an out-of-band application built-in to the calling device, for example, this is the same check performed in Step 1: does the calling device hold a private key, one corresponding to a STIR certificate, that can sign for the originating identity?

Step 3: The authentication service MUST acquire the public encryption key of the destination, which will be used to encrypt the PASSporT (see [Section 11](#)). It MUST also discover (see [Section 10](#)) the CPS associated with the destination. The authentication service may already have the encryption key and destination CPS cached, or may need to query a service to acquire the key. Note that per [Section 7.5](#) the authentication service may also need to acquire a token for PASSporT storage from the CPS upon CPS discovery. It is anticipated that the discovery mechanism (see [Section 10](#)) used to find the appropriate CPS will also find the proper key server for the public key of the destination. In some cases, a destination may have multiple public encryption keys associated with it. In that case, the authentication service MUST collect all of those keys.

Step 4: The authentication service MUST create the PASSporT object. This includes acquiring the system time to populate the "iat" claim, and populating the "orig" and "dest" claims as described in [[RFC8225](#)]. The authentication service MUST then encrypt the PASSporT. If in Step 3 the authentication service discovered multiple public keys for the destination, it MUST create one encrypted copy for each public key it discovered.

Finally, the authentication service stores the encrypted PASSporT(s) at the CPS discovered in Step 3. Only after that is completed should any call be initiated. Note that a call might be initiated over SIP, and the authentication service would place the same PASSporT in the Identity header field value of the SIP request - though SIP would carry a cleartext version rather than an encrypted version sent to the CPS. In that case, out-of-band would serve as a fallback mechanism in case the request was not conveyed over SIP end-to-end.

Also, note that the authentication service MAY use a compact form of the PASSporT for a SIP request, whereas the version stored at the CPS MUST always be a full form PASSporT.

8.2. Verification Service

When a call arrives, an out-of-band verification service performs steps similar to those defined in [[RFC8224](#)] with some exceptions:

Step 1: The verification service contacts the CPS and requests all current PASSporTs for its destination number; or alternatively it may receive PASSporTs through a push interface from the CPS in some deployments. The verification service MUST then decrypt all PASSporTs using its private key. Some PASSporTs may not be decryptable for any number of reasons: they may be intended for a different verification service, or they may be "dummy" values inserted by the CPS for privacy purposes. The next few steps will narrow down the set of PASSporTs that the verification service will examine from that initial decryptable set.

Step 2: The verification service MUST determine if any "ppt" extensions in the PASSporTs are unsupported. It takes only the set of supported PASSporTs and applies the next step to them.

Step 3: The verification service MUST determine if there is an overlap between the calling party number number presented in call signaling and the "orig" field of any decrypted PASSporTs. It takes the set of matching PASSporTs and applies the next step to them.

Step 4: The verification service MUST determine if the credentials that signed each PASSporT are valid, and if the verification service trusts the CA that issued the credentials. It takes the set of trusted PASSporTs to the next step.

Step 5: The verification service MUST check the freshness of the "iat" claim of each PASSporT. The exact interval of time that determines freshness is left to local policy. It takes the set of fresh PASSporTs to the next step.

Step 6: The verification service MUST check the validity of the signature over each PASSporT, as described in [[RFC8225](#)].

Finally, the verification service will end up with one or more valid PASSporTs corresponding to the call it has received. This document does not prescribe any particular treatment of calls that have valid PASSporTs associated with them. The handling of the message after the verification process depends on how the verification service is implemented and on local policy. However, 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 UA implementations.

8.3. Gateway Placement Services

The STIR out-of-band mechanism also supports the presence of gateway placement services, which do not create PASSporTs themselves, but instead take PASSporTs out of signaling protocols and store them at a CPS before gatewaying to a protocol that cannot carry PASSporTs itself. For example, a SIP gateway that sends calls to the PSTN could receive a call with an Identity header field, extract a PASSporT from the Identity header field, and store that PASSporT at a CPS.

To place a PASSporT at a CPS, a gateway **MUST** perform Step 3 of [Section 8.1](#) above: that is, it must discover the CPS and public key associated with the destination of the call, and may need to acquire a PASSporT storage token (see [Section 6.1](#)). Per Step 3 this may entail discovering several keys. The gateway then collects the in-band PASSporT(s) from the in-band signaling, encrypts the PASSporT(s), and stores them at the CPS.

A similar service could be performed by a gateway that retrieves PASSporTs from a CPS and inserts them into signaling protocols that support carrying PASSporTs in-band. This behavior may be defined by future specifications.

9. Example HTTPS Interface to the CPS

As a rough example, we show a Call Placement Service implementation here which uses a REST API to store and retrieve objects at the CPS. The calling party stores the PASSporT at the CPS prior to initiating the call; the PASSporT is stored at a location at the CPS that corresponds to the called number. Note that it is possible for multiple parties to be calling a number at the same time, and that for called numbers such as large call centers, many PASSporTs could legitimately be stored simultaneously, and it might prove difficult to correlate these with incoming calls.

Assume that an authentication service has created the following PASSporT for a call to the telephone number 2.222.555.2222 (note that these are dummy values):


```
eyJhbGciOiJFUzI1NiIsInR5cCI6InBhc3Nwb3J0IiwieDV1IjoiaHR0cHM6Ly9j
ZXJ0LmV4YW1wbGUub3JnL3Bhc3Nwb3J0LmNlciJ9.eyJkZXN0Ijp7InVyaSI6WyJz
aXA6YWxpY2VhbnBhbmV4b3N5b20iXX0sImldhdCI6IjE0NDMyMDgzNDUiLCJvcmlnI
jp7InRuIjoiaMTIiXNTU1NTEyMTIifX0.rq3pjT1hoRwakEGjHCnWSwUnshd0-zJ6F1
V0gFWSjHBr8Qjpjlk-cpFYpFYsojNCpTz03QfP0lckGaS6hEck7w
```

Through some discovery mechanism (see [Section 10](#)), the authentication service discovers the network location of a web service that acts as the CPS for 2.222.555.2222. Through the same mechanism, we will say that it has also discovered one public encryption key for that destination. It uses that encryption key to encrypt the PASSporT, resulting in the encrypted PASSporT:

```
rLWuoTpvBvWSHmV1AvVfVaE5pPV6Va0up3Ajo3W0VvjvrQI1VwbvnUE0pUZ6Yl9w
MKW0YzI4LJ1joTHho3WaY30up3Ajo3W0YzAypvW9rlWxMKA0Vwc7VaIlNFV6JlWm
nKN6LJkcl2INMKuu0K0fMF5wo20vKK0fVzyuqPV6VwR0AQZLZQtmAQHvYPWipzyaV
wc7VaEhVwbvZGVkAGH1AGRlZGVvsK0ed3cwG1ubEjnxRTwUPaJFjHafuq0-mW6S1
IBtSJFwU0e8Dwcwyx-pcSLcSLfbwAPcGmB3DsCBypxTnF6uRpx7j
```

Having concluded the numbered steps in [Section 8.1](#), including acquiring any token (per [Section 6.1](#)) needed to store the PASSporT at the CPS, the authentication service then stores the encrypted PASSporT:

```
POST /cps/2.222.555.2222/ppts HTTP/1.1
Host: cps.example.com
Content-Type: application/passport
```

```
rLWuoTpvBvWSHmV1AvVfVaE5pPV6Va0up3Ajo3W0VvjvrQI1VwbvnUE0pUZ6Yl9w
MKW0YzI4LJ1joTHho3WaY30up3Ajo3W0YzAypvW9rlWxMKA0Vwc7VaIlNFV6JlWm
nKN6LJkcl2INMKuu0K0fMF5wo20vKK0fVzyuqPV6VwR0AQZLZQtmAQHvYPWipzyaV
wc7VaEhVwbvZGVkAGH1AGRlZGVvsK0ed3cwG1ubEjnxRTwUPaJFjHafuq0-mW6S1
IBtSJFwU0e8Dwcwyx-pcSLcSLfbwAPcGmB3DsCBypxTnF6uRpx7j
```

The web service assigns a new location for this encrypted PASSporT in the collection, returning a 201 OK with the location of /cps/2.222.222.2222/ppts/ppt1. Now the authentication service can place the call, which may be signaled by various protocols. Once the call arrives at the terminating side, a verification service contacts its CPS to ask for the set of incoming calls for its telephone number (2.222.222.2222).

```
GET /cps/2.222.555.2222/ppts
Host: cps.example.com
```

This returns to the verification service a list of the PASSporTs currently in the collection, which currently consists of only

/cps/2.222.222.2222/ppts/ppt1. The verification service then sends a new GET for /cps/2.222.555.2222/ppts/ppt1/ which yields:

```
HTTP/1.1 200 OK
Content-Type: application/passport
Link: <https://cps.example.com/cps/2.222.555.2222/ppts>
```

```
rlWuoTpvBvWSHmV1AvVfVaE5pPV6Va0up3Ajo3W0VvjvrQI1VwbvnUE0pUZ6Yl9w
MKW0YzI4LJ1joTHho3WaY30up3Ajo3W0YzAypvW9rlWxMKA0Vwc7VaIlNFV6JlWm
nKN6LJkcL2INMKuu0K0fMF5wo20vKK0fVzyuqPV6VwR0AQZLZQtmAQHvYPWipzyaV
wc7VaEhVwbvZGVkAGH1AGRLZGVvsK0ed3cwG1ubEjnxRTwUPaJFjHafuq0-mW6S1
IBtSJFwU0e8Dwcwyx-pcSLcSLfbwAPcGmB3DsCBypxTnF6uRpx7j
```

That concludes Step 1 of [Section 8.2](#); the verification service then goes on to the next step, processing that PASSporT through its various checks. A complete protocol description for CPS interactions is left to future work.

10. CPS Discovery

In order for the two ends of the out-of-band dataflow to coordinate, they must agree on a way to discover a CPS and retrieve PASSporT objects from it based solely on the rendezvous information available: the calling party number and the called number. Because the storage of PASSporTs in this architecture is indexed by the called party number, it makes sense to discover a CPS based on the called party number as well. There are a number of potential service discovery mechanisms that could be used for this purpose. The means of service discovery may vary by use case.

Although the discussion above is written largely in terms of a single CPS, having a significant fraction of all telephone calls result in storing and retrieving PASSporTs at a single monolithic CPS has obvious scaling problems, and would as well allow the CPS to gather metadata about a very wide set of callers and callees. These issues can be alleviated by operational models with a federated CPS; any service discovery mechanism for out-of-band STIR should enable federation of the CPS function. Likely models include ones where a carrier operates one or more CPS instances on behalf of its customers, enterprises run a CPS instance on behalf of their PBX users, or where third-party service providers offer a CPS as a cloud service.

Some service discovery possibilities under consideration include the following:

For some deployments in closed (e.g. intranetwork) environments, the CPS location can simply be provisioned in implementations, obviating the need for a discovery protocol.

If a credential lookup service is already available (see [Section 11](#)), the CPS location can also be recorded in the callee's credentials; an extension to [\[RFC8226\]](#) could for example provide a link to the location of the CPS where PASSporTs should be stored for a destination.

There exist a number of common directory systems that might be used to translate telephone numbers into the URIs of a CPS. ENUM [\[RFC6116\]](#) is commonly implemented, though no "golden root" central ENUM administration exists that could be easily reused today to help the endpoints discover a common CPS. Other protocols associated with queries for telephone numbers, such as the TeRI [\[I-D.ietf-modern-teri\]](#) protocol, could also serve for this application.

Another possibility is to use a single distributed service for this function. VIPR [\[I-D.jennings-vipr-overview\]](#) proposed a RELOAD [\[RFC6940\]](#) usage for telephone numbers to help direct calls to enterprises on the Internet. It would be possible to describe a similar RELOAD usage to identify the CPS where calls for a particular telephone number should be stored. One advantage that the STIR architecture has over VIPR is that it assumes a credential system that proves authority over telephone numbers; those credentials could be used to determine whether or not a CPS could legitimately claim to be the proper store for a given telephone number.

This document does not prescribe any single way to do service discovery for a CPS; it is envisioned that initial deployments will provision the location of the CPS at the Authentication Service and Verification Service.

[11. Encryption Key Lookup](#)

In order to encrypt a PASSporT (see [Section 6.1](#)), the caller needs access to the callee's public encryption key. Note that because STIR uses ECDSA for signing PASSporTs, the public key used to verify PASSporTs is not suitable for this function, and thus the encryption key must be discovered separately. This requires some sort of directory/lookup system.

Some initial STIR deployments have fielded certificate repositories so that verification services can acquire the signing credentials for PASSporTs, which are linked through a URI in the "x5u" element of the

PASSporT. These certificate repositories could clearly be repurposed for allowing authentication services to download the public encryption key for the called party - provided they can be discovered by calling parties. This document does not specify any particular discovery scheme, but instead offers some general guidance about potential approaches.

It is a desirable property that the public encryption key for a given party be linked to their STIR credential. We therefore propose that an ECDH [RFC7748] public-private key pair be generated for a subcert [I-D.ietf-tls-subcerts] of the STIR credential. That subcert could be looked up along with the STIR credential of the called party. Further details of this subcert, and the exact lookup mechanism involved, are deferred for future protocol work.

Obviously, if there is a single central database that the caller and callee each access in real time to download the other's keys, then this represents a real privacy risk, as the central key database learns about each call. A number of mechanisms are potentially available to mitigate this:

- Have endpoints pre-fetch keys for potential counterparties (e.g., their address book or the entire database).

- Have caching servers in the user's network that proxy their fetches and thus conceal the relationship between the user and the keys they are fetching.

Clearly, there is a privacy/timeliness tradeoff in that getting up-to-date knowledge about credential validity requires contacting the credential directory in real-time (e.g., via OCSP [RFC2560]). This is somewhat mitigated for the caller's credentials in that he can get short-term credentials right before placing a call which only reveals his calling rate, but not who he is calling. Alternately, the CPS can verify the caller's credentials via OCSP, though of course this requires the callee to trust the CPS's verification. This approach does not work as well for the callee's credentials, but the risk there is more modest since an attacker would need to both have the callee's credentials and regularly poll the database for every potential caller.

We consider the exact best point in the tradeoff space to be an open issue.

12. Acknowledgments

The ideas in this document come out of discussions with Richard Barnes and Cullen Jennings. We'd also like to thank Russ Housley, Chris Wendt, Eric Burger, Mary Barnes, Ben Campbell, Ted Huang, Jonathan Rosenberg and Robert Sparks for helpful suggestions.

13. IANA Considerations

This memo includes no request to IANA.

14. Security Considerations

This entire document is about security, but the detailed security properties will vary depending on how the framework is applied and deployed. General guidance for dealing with the most obvious security challenges posed by this framework is given in [Section 7.3](#) and [Section 7.4](#), along proposed solutions for problems like denial-of-service attacks or traffic analysis against the CPS.

Although there are considerable security challenges associated with widespread deployment of a public CPS, those must be weighed against the potential usefulness of a service that delivers a STIR assurance without requiring the passage of end-to-end SIP. Ultimately, the security properties of this mechanism are at least comparable to in-band STIR: the substitution attack documented in [Section 7.4](#) could be implemented by any in-band SIP intermediary or eavesdropper who happened to see the PASSporT in transit, say, and launch its own call with a copy of that PASSporT to race against the original to the destination.

15. Informative References

[I-D.ietf-modern-teri]

Peterson, J., "An Architecture and Information Model for Telephone-Related Information (TeRI)", [draft-ietf-modern-teri-00](#) (work in progress), July 2018.

[I-D.ietf-stir-passport-divert]

Peterson, J., "PASSporT Extension for Diverted Calls", [draft-ietf-stir-passport-divert-06](#) (work in progress), July 2019.

[I-D.ietf-tls-subcerts]

Barnes, R., Iyengar, S., Sullivan, N., and E. Rescorla, "Delegated Credentials for TLS", [draft-ietf-tls-subcerts-04](#) (work in progress), July 2019.

[I-D.jennings-vipr-overview]

Barnes, M., Jennings, C., Rosenberg, J., and M. Petit-Huguenin, "Verification Involving PSTN Reachability: Requirements and Architecture Overview", [draft-jennings-vipr-overview-06](#) (work in progress), December 2013.

[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>>.

[RFC2560] Myers, M., Ankney, R., Malpani, A., Galperin, S., and C. Adams, "X.509 Internet Public Key Infrastructure Online Certificate Status Protocol - OCSP", [RFC 2560](#), DOI 10.17487/RFC2560, June 1999, <<https://www.rfc-editor.org/info/rfc2560>>.

[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, <<https://www.rfc-editor.org/info/rfc3261>>.

[RFC5636] Park, S., Park, H., Won, Y., Lee, J., and S. Kent, "Traceable Anonymous Certificate", [RFC 5636](#), DOI 10.17487/RFC5636, August 2009, <<https://www.rfc-editor.org/info/rfc5636>>.

[RFC6116] Bradner, S., Conroy, L., and K. Fujiwara, "The E.164 to Uniform Resource Identifiers (URI) Dynamic Delegation Discovery System (DDDS) Application (ENUM)", [RFC 6116](#), DOI 10.17487/RFC6116, March 2011, <<https://www.rfc-editor.org/info/rfc6116>>.

[RFC6940] Jennings, C., Lowekamp, B., Ed., Rescorla, E., Baset, S., and H. Schulzrinne, "REsource LOcation And Discovery (RELOAD) Base Protocol", [RFC 6940](#), DOI 10.17487/RFC6940, January 2014, <<https://www.rfc-editor.org/info/rfc6940>>.

[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>>.

[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>>.

- [RFC7748] Langley, A., Hamburg, M., and S. Turner, "Elliptic Curves for Security", [RFC 7748](#), DOI 10.17487/RFC7748, January 2016, <<https://www.rfc-editor.org/info/rfc7748>>.
- [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, February 2018, <<https://www.rfc-editor.org/info/rfc8224>>.
- [RFC8225] Wendt, C. and J. Peterson, "PASSporT: Personal Assertion Token", [RFC 8225](#), DOI 10.17487/RFC8225, February 2018, <<https://www.rfc-editor.org/info/rfc8225>>.
- [RFC8226] Peterson, J. and S. Turner, "Secure Telephone Identity Credentials: Certificates", [RFC 8226](#), DOI 10.17487/RFC8226, February 2018, <<https://www.rfc-editor.org/info/rfc8226>>.
- [RFC8446] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", [RFC 8446](#), DOI 10.17487/RFC8446, August 2018, <<https://www.rfc-editor.org/info/rfc8446>>.

Authors' Addresses

Eric Rescorla
Mozilla

Email: ekr@rtfm.com

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

Email: jon.peterson@team.neustar