

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

J. Peterson
T. McGarry
NeuStar, Inc.
July 6, 2015

Modern Problem Statement, Use Cases, and Framework
draft-peterson-modern-problems-01.txt

Abstract

The functions of the public switched telephone network (PSTN) are gradually migrating to the Internet. This is generating new requirements for many mechanisms used on the PSTN, including telephone numbers (TNs). TNs no longer serve simply as telephone routing addresses, they are now identifiers which may be used by Internet-based services for a variety of purposes including session establishment, identity verification, and service enablement. This problem statement examines how the existing tools for allocating and managing telephone numbers do not align with the needs of the Internet environment and proposes a framework for Internet-based services relying on TNs.

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. Problem Statement	2
2. Actors	4
3. Framework	5
4. Use Cases	6
4.1. Acquiring Telephone Numbers	6
4.1.1. CSP Acquires Numbers from Registry	6
4.1.2. User/Delegate Acquires TNs from CSP	7
4.1.3. User Acquires TNs from a Delegate	8
4.1.4. User Acquires Numbers from Registry	8
4.2. Accessing Numbering Information	8
4.2.1. Service Information Access	8
4.2.2. Privileged Access for Government Entities	9
4.3. Service Management for Numbers	9
4.3.1. Updating Service Information	9
4.3.2. Updating Administrative Information	10
4.3.3. Changing the CSP for an Existing Communications Service	10
4.3.4. Terminating a Service	10
5. Distributed Registries and Data Stores	11
6. Acknowledgments	11
7. IANA Considerations	11
8. Security Considerations	11
9. Informative References	12
Authors' Addresses	13

1. Problem Statement

The challenges of utilizing telephone numbers (TNs) on the Internet has been known for some time. Internet telephony provided the main use case for routing telephone numbers on the Internet in a manner similar to how calls are routed in the public switched telephone network (PSTN). As the Internet had no service for discovering the endpoints associated with telephone numbers, ENUM [3] created a DNS-based mechanism for resolving TNs in an IP environment, by defining procedures for translating TNs into URIs for use by protocols such as SIP [2]. Originally, it was envisioned that ENUM would be deployed as a global hierarchical service, though in practice, it has only been deployed piecemeal by various parties. Most notably, ENUM is used as an internal network function, and is hardly used between

service provider networks. The original ENUM concept of a single root, `el64.arpa`, proved to be politically challenging, and less centralized models have thus flourished.

Subsequently, the DRINKS [4] framework showed ways that authorities might provision information about telephone numbers at an ENUM service or similar Internet-based directory. These technologies have generally tried to preserve the features and architecture familiar from the PSTN numbering environment.

Telephone numbering, however, has long been transitioning away from a provider-centric model towards a user-centric model. Number portability has been implemented in many countries, and the right of a user to choose and change their service provider while retaining their TN is widely acknowledged now. However, TN administration processes rooted in PSTN technology and policies dictate that this be an exception process fraught with problems and delays. Thanks to the increasing sophistication of consumer mobile devices, users now associate telephone numbers with many applications other than telephony. Ideally the user would have full control of their TN and would drive the porting process on their own rather than rely on complex and time consuming back office processes among multiple service providers.

Most TNs today are assigned to specific geographies, at both an international level and within national numbering plans. This has shaped the way that service providers interconnect, as well as how telephone numbers are routed and administered: the PSTN was carefully designed to delegate switching intelligence geographically. In interexchange carrier routing in North America, for example, calls to a particular TN are often handed off to the terminating service provider close to the geography where that TN is assigned. But the overwhelming success of mobile telephones has increasingly eroded the connection between numbers and regions. Furthermore, the topology of IP networks is not anchored to geography in the same way that the telephone network is. In an Internet environment, establishing a network architecture for routing telephone numbers would depend little on geography. Adapting telephone numbers to the Internet requires more security, richer datasets and more complex query and response capabilities than previous efforts have provided.

With the PSTN well on its way to transitioning to an all IP network, and TNs showing no signs of sunseting as a resource, it is time to address the issues of routing, management and administration of TNs in an IP environment. This document will create a common understanding of the problem statement related to TNs in an IP environment and help develop a vision for how to create IP-based mechanisms for TNs. It will be important to acknowledge that there

are various international and national policies and processes related to TNs, and any solutions need to be flexible enough to account for these variations.

2. Actors

The following actors are defined in this document:

Numbering Authority: A regulatory body within a country that manages that country's telephone numbers. The numbering authority decides national numbering policy, including what telephone numbers can be allocated, and which are reserved.

Registry: An entity that administers the allocation of telephone numbers based on a numbering authority's policies. Numbering authorities can act as the registries themselves, or they can outsource the function to other entities. A registry can act as a sole authoritative entity for a numbering authority, or there can also be multiple registries that manage the same telephone numbers and synchronize with each other.

Communication Service Provider: A provider of communications services to users, where those services can be identified by telephone numbers. This includes both traditional telephone carriers or enterprises as well as service providers with no presence on the PSTN who use telephone numbers. This framework does not assume that any single CSP provides all the communications service related to a TN.

Service Enabler: An entity that works with CSPs to enable communication service to a user; perhaps a vendor, or third-party integrator.

User: An individual reachable through a communications service; usually a customer of a communication service provider who uses telephone numbers to reach and identify services. Sophisticated users may also act as their own CSPs.

Government Entity: An entity that, due to legal powers deriving from national policy, has privileged access to information about number administration under certain conditions.

Note that a given entity may act in one or more of the roles above. An entity acting as a Communications Service Provider, Service Enabler, or User can also be said to have a relationship to the registry of either an assignee or delegate:

Assignee.: An entity that is assigned the telephone number by the registry. There is a direct relationship between the registry and the assignee.

Delegate: An entity that is delegated a telephone number from an assignee or another delegate for assignment or delegation to others. A delegate is not the assignee or the user.

Note that although numbering authorities are listed as actors, they are unlikely to actually participate in the protocol flows themselves.

3. Framework

The framework outlined in this document requires three Internet-based mechanisms for managing and resolving telephone numbers (TNs) in an IP environment. These mechanisms will likely reuse existing protocols for sharing structured data; it is unlikely that new protocol development work will be required, though new information models specific to the data itself will be a major focus of framework development. Likely candidates for reuse here include work done in DRINKS and WEIRDS, as well as the TeRQ [12] framework.

These protocol mechanisms are scoped in a way that makes them likely to apply to a broad range of future policies for number administration. It is not the purpose of this framework to dictate number policy, but instead to provide tools that will work with policies as they evolve going forward. These mechanisms therefore do not assume that number administration is centralized, nor that number "ownership" is restricted to any privileged service providers, though these tools must and will work in environments with those properties.

The three mechanisms are:

Acquisition: a protocol mechanism to enable users or CSPs to acquire TNs from authorities, including an enrollment process for the individuals and entities that manage TNs.

Management: a protocol mechanism for users to associate data with TNs at a CSP.

Retrieval: a protocol mechanism for service providers, users, and government entities to retrieve data about TNs from either an authority or a CSP.

The acquisition mechanism will enable actors to acquire telephone numbers for use with a communications service. The acquisition mechanism will provide a means for either a user or a CSP to request

numbering resources from an authority, either on a number-by-number basis, or as inventory blocks. The authority who grants numbering resources to a user will retain metadata about the assignment, including the responsible organization or individual to whom numbers have been assigned. In the DNS environment, an authority thus might be analogous to either a registrar or a reseller of names, though obvious hierarchical domain names do not have a comparable inventory situation to telephone numbers.

The management mechanism will let actors provision data associated with telephone numbers at CSPs. If a user owns a telephone number, they may select a CSP to provide a particular service associated with the number, or a CSP may own a number, and effectively rent these to users. In either case, a user needs a mechanism to provision data associated with the number at a CSP.

The resolution mechanism will enable actors to learn information about telephone numbers, typically by sending a request to a CSP. For some information, an actor may need to send a request to a numbering authority rather than a CSP. Different parties may be authorized to receive different information about telephone numbers.

4. Use Cases

The high-level use cases in this section will provide an overview of the expected operation of the three protocols in the MODERN problem space.

4.1. Acquiring Telephone Numbers

There are various scenarios for how TNs can be acquired by the relevant actors; registry, CSP, delegate, service enabler, and user. The registry perform its functions as defined by the national authority, so the national authority does not participate in the protocol flows in this section.

4.1.1. CSP Acquires Numbers from Registry

Through some out-of-band business process, a CSP develops a relationship with a Registry. The Registry has a profile of the CSP and what qualifications they possess for requesting TNs. The CSP may then request TNs from within a specific pool of numbers in the authority of the Registry; such as region, mobile, wireline, tollfree, etc. The Registry must authenticate and authorize the CSP, and then either grant or deny a request. When an assignment occurs, the registry stores information related to the assignment including the resource and the assignee, and removes the specific TN(s) from the pool of those that are available for assignment. As a part of

the assignment process, the Registry provides credentials (for example, STIR certificates [13]) to the CSP to be used to prove the assignment for future transactions

Before it is eligible to receive number assignments, per the policy of a national authority, the CSP may need to have submitted (again, through some out-of-band process) additional qualifying information such as current utilization rate or a demand forecast.

There are two scenarios under which a CSP requests resources; they are requesting inventory, or they are requesting for a specific user or delegate. If they are requesting for a user or delegate they may need to register information about the user or delegate with the Registry. Examples of user and delegate information could be contact information that may be required by Government Entities, or some forms of service information. Such data could be provided to the Registry or the Registry could be provided with an address where that data can be accessed. Such an address could be part of the CSP profile with the Registry.

4.1.2. User/Delegate Acquires TNs from CSP

A User or Delegate creates or has a relationship with the CSP, and subscribes to a communications service which includes the use of a telephone number. The CSP first collects and stores profile data about the User or Delegate. The CSP then activates the User or Delegate on their network and creates any necessary service data to enable interoperability with other CSPs. The CSP could also update public or privileged databases accessible by other Actors. The CSP provides a credential to the User or Delegate (for example, a STIR certificate [13]) to prove the assignment for future transactions. The credential could be delegated from the one provided by the Registry to the CSP in a continuing the chain of assignment.

The CSP could assign a TN from its existing inventory or it could acquire it from the Registry as part of the assignment process. If it assigns it from its existing inventory it would remove the specific TN from the pool of those available for assignment. It may also update the registry about the assignment so the registry has current utilization data. If TNs are assigned to a Delegate for use as inventory to be assigned to Users, the Delegate may need to provide utilization data to the Registry, either directly or through the CSP.

4.1.3. User Acquires TNs from a Delegate

This follows the process in Section 4.1.2, as it should be similar to how a User acquires TNs from a CSP. In this case, the Delegate would be performing functions done by the CSP, e.g., providing credentials, updating the Registry, and so on.

4.1.4. User Acquires Numbers from Registry

This follows the process in Section 4.1.1, as it should be similar to how a CSP acquires TNs from a Registry. In this case, the user must establish some business relationship directly to a registry, perhaps similarly to how such functions are conducted today when users purchase domain names. TNs assigned to a user are always considered assigned by the Registry, not inventory.

In this use case, after receiving a number assignment from the Registry, a User will then obtain communications service from a CSP, and provide to the CSP the TN to be used for that service. The CSP will associate service information for that TN, e.g., service address, and make it available to other CSPs to enable interoperability.

4.2. Accessing Numbering Information

Telephone numbering information will generally fall into two categories; administrative information and service information. Administrative information includes TN status, service provider, contact data, etc. and typically does not require real-time performance. Service information includes addressing data, feature capabilities, etc. and typically does require real-time performance.

Telephone numbering data can be stored at the Registry or at the CSP that holds the information. The address for accessing the information would need to be available to others to enable access and interoperability. For example, if the data is held by the Registry, a URL for accessing that data could be published to those that have access to the Registry. The Registry could allow and restrict access to specific information based on the identity of requestor. If the data is held by a CSP, the Registry could host an address for each TN that references the correct CSP, and the CSP would allow or restrict access based on the requestor.

4.2.1. Service Information Access

A gateway receives a call for a telephone number. That telephone number is assigned to a CSP, who has delegated to the number to a User. The gateway wants to reach the User through an Internet

communications endpoint. It therefore send a query to the Registry responsible for the numbering space that the telephone number resides in. The Registry proxies or redirects the request to the CSP that has been assigned the number. The CSP returns Internet endpoint information for that number to the gateway, possibly after making an authorization decision.

In an alternative use case, the CSP might provision the Registry with endpoint service information for the telephone number, or the CSP might have delegated to the User the responsibility for provisioning this information with the Registry.

4.2.2. Privileged Access for Government Entities

In this case, a Government Entity wishes to access information about a particular User, who subscribes to a communications service. The entity that operates the Registry on behalf of the National Authority in this case has some pre-defined relationship with the Government Entity. When the CSP acquired TNs from the National Authority, it was a condition of that assignment that the CSP provide access for Government Entities to telephone numbering data when certain conditions apply. The required data may reside either in the CSP or in the Registry.

For a case where the CSP delegates a number to the User, the CSP might provision the Registry with information relevant to the User. At such a time as the Government Entity needs information about that User, the Government Entity may contact the Registry or CSP to acquire the necessary data. The interfaces necessary for this will be the same as those described in Section 4.2; the Government Entity will be authenticated, and an authorization decision will be made by the Registry or CSP under the policy dictates established by the National Authority.

4.3. Service Management for Numbers

The use cases in this section describe ways that numbering data, including administrative information and service information, might be updated at the CSP or Registry after a number has been initially assigned and provisioned.

4.3.1. Updating Service Information

A CSP handles all users in a large region through a central set of proxy servers. They provision a URL pointing to that service in the Registry. After a business transition, the CSP wants to point the service to a new URL. The CSP therefore sends a provisioning update to the Registry.

While some similar use cases may apply to individual Users, it is anticipated that for the most part these lower-level service information changes would be communicated via existing protocols (like the baseline [2] SIP REGISTER method) rather than through any interfaces defined by MODERN.

4.3.2. Updating Administrative Information

A User who subscribes to a communications service changes their postal address, moving from one location in the country to another. At this time, the User informs the CSP. The CSP updates its own records, and send an update to the Registry as well, as the National Authority in this case requires that the CSP notify the Registry of changes in the contact information associated with numbering resources.

4.3.3. Changing the CSP for an Existing Communications Service

A User who subscribes to a communications service, and received their TN from that CSP, wishes to retain the same TN but move their service to a different CSP.

Depending on the policies set by the National Authority, it might be the responsibility of either the old or new CSP to initiate the transition process. The new CSP will then provision the registry with the new service and administrative information associated with the CSP, though much of the administrative information relating to the User may remain the same through this transition. The CSP will perform other functions described above in Section 4.1.2.

At this time, the old CSP will undo any delegations to the User, including invalidating any cryptographic credentials (e.g. STIR certificates [13]) previously granted to the user. Any routing or service information maintained by the CSP must be removed, and similarly, the CSP must delete any such information it provisioned in the Registry.

[TBD - more on the case where multiple CSPs provide services for a given TN, and only one service is "ported" to a new CSP?]

4.3.4. Terminating a Service

This use case is very similar to that in Section 4.3.3. A User who subscribes to a communications service, and received their TN from the CSP, wishes to terminate their service. At this time, the CSP will undo any delegations to the User, including invalidating any cryptographic credentials (e.g. STIR certificates [13]) previously granted to the user. Any routing or service information maintained

by the CSP must be removed, and similarly, the CSP must delete any such information it provisioned in the Registry.

In an alternative use case, a User who received their own TN assignment directly from the Registry terminates their service with a CSP. At this time, the User might terminate their assignment from the Registry, and return the number to the Registry for re-assignment. Alternatively, they could retain the number and elect to assign it to some other service at a later time.

5. Distributed Registries and Data Stores

It is possible to create a distributed Registry or distributed Data Stores for the administrative and service information associated with a TN.

In a distributed Registry there would be multiple duplicate copies of the Registry data. A CSP or User would interact with one Registry and that Registry would be responsible for initiating updates to all other Registries when there is a change. The challenge is to ensure that there are no clashes, e.g., two Registries assigning the same TN to two different CSPs.

Similarly multiple entities can maintain duplicate copies of administrative and service data associated with TNs. For example, when a CSP enables service for a User they can initiate an update of the service address to multiple other data stores managed by other service providers. This may not be the best solution for User contact data.

[More TBD]

6. Acknowledgments

We would like to thank Henning Schulzrinne for his contributions to this problem statement and framework.

7. IANA Considerations

This memo includes no request to IANA.

8. Security Considerations

TBD.

9. Informative References

- [1] Peterson, J. and C. Jennings, "Enhancements for Authenticated Identity Management in the Session Initiation Protocol (SIP)", RFC 4474, August 2006.
- [2] 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.
- [3] Bradner, S., Conroy, L., and K. Fujiwara, "The E.164 to Uniform Resource Identifiers (URI) Dynamic Delegation Discovery System (DDDS) Application (ENUM)", RFC 6116, March 2011.
- [4] Channabasappa, S., "Data for Reachability of Inter-/Intra-Network SIP (DRINKS) Use Cases and Protocol Requirements", RFC 6461, January 2012.
- [5] Watson, M., "Short Term Requirements for Network Asserted Identity", RFC 3324, November 2002.
- [6] 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.
- [7] Hoffman, P. and J. Schlyter, "The DNS-Based Authentication of Named Entities (DANE) Transport Layer Security (TLS) Protocol: TLSA", RFC 6698, August 2012.
- [8] Elwell, J., "Connected Identity in the Session Initiation Protocol (SIP)", RFC 4916, June 2007.
- [9] Schulzrinne, H., "The tel URI for Telephone Numbers", RFC 3966, December 2004.
- [10] Rosenberg, J. and C. Jennings, "The Session Initiation Protocol (SIP) and Spam", RFC 5039, January 2008.
- [11] Peterson, J., Jennings, C., and R. Sparks, "Change Process for the Session Initiation Protocol (SIP) and the Real-time Applications and Infrastructure Area", BCP 67, RFC 5727, March 2010.

- [12] Peterson, J., "A Framework and Information Model for Queries about Telephone-Related Queries (TeRQ)", draft-peterson-terq-03 (work in progress), February 2013.
- [13] Peterson, J., "Secure Telephone Identity Credentials: Certificates", draft-ietf-stir-certificates-01 (work in progress), March 2015.
- [14] 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.
- [15] Rosenberg, J. and H. Schulzrinne, "Session Initiation Protocol (SIP): Locating SIP Servers", RFC 3263, June 2002.

Authors' Addresses

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

Email: jon.peterson@neustar.biz

Tom McGarry
Neustar, Inc.
1800 Sutter St Suite 570
Concord, CA 94520
US

Email: tom.mcgarry@neustar.biz

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: April 21, 2016

J. Peterson
Neustar, Inc.
October 19, 2015

A Framework and Information Model for Telephone-Related Information
(TeRI)
draft-peterson-modern-teri-00

Abstract

As telephone services migrate to the Internet, Internet applications require tools to access and manage information about telephone numbers. This document specifies a protocol-independent framework and information model for managing service and administration data associated with telephone numbers.

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 April 21, 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.

This document may contain material from IETF Documents or IETF Contributions published or made publicly available before November 10, 2008. The person(s) controlling the copyright in some of this material may not have granted the IETF Trust the right to allow modifications of such material outside the IETF Standards Process. Without obtaining an adequate license from the person(s) controlling the copyright in such materials, this document may not be modified outside the IETF Standards Process, and derivative works of it may not be created outside the IETF Standards Process, except to format it for publication as an RFC or to translate it into languages other than English.

Table of Contents

1. Terminology	3
2. Motivation	3
3. The Information Model	4
3.1. Record Elements	5
3.1.1. Identifier	5
3.1.2. Authority	5
3.1.3. Contact	5
3.1.4. Subject	5
3.1.5. Service	6
3.1.6. Signature	6
3.2. Element Value Types	6
3.2.1. Service Types	6
3.2.2. Public Key Type	8
3.2.3. Contact Type	8
3.2.4. Expiry Type	8
3.2.5. Priority Type	8
3.2.6. Record Identifier Type	8
3.2.7. Signature	8
3.2.8. Extension Type	8
4. Operations	9
4.1. Common to All Operations	9
4.1.1. Requests	9
4.1.2. Responses	11
4.2. The Acquisition Operation	11
4.3. The Management Operation	12
4.4. The Retrieval Operation	12
4.5. Common Attributes	12
4.5.1. Administrative Attributes	13
4.5.2. Service Attributes	13
5. Implementing Operations	13
5.1. Transport Independence	14

5.2. Bindings	14
5.3. Encodings	15
5.4. Profiles	16
6. Security Considerations	16
7. IANA Considerations	16
8. Acknowledgements	17
9. Informative References	17
Author's Address	20

1. Terminology

In this document, the key words "MAY", "MUST", "MUST NOT", "SHOULD", and "SHOULD NOT", are to be interpreted as described in [RFC2119].

2. Motivation

Telephone numbers remain the worldwide standard identifier for routing calls and text messages over the Public Switched Telephone Network (PSTN). Increasingly, real-time communications is migrating to the Internet, and bringing telephone numbers with it. As identifiers, however, telephone numbers differ fundamentally from those commonly used by Internet applications. Email, the web and native Voice over IP (VoIP) systems such as SIP ([RFC3261]) typically use identifiers that rely on the Domain Name System (DNS) to resolve a domain portion of the identifier to a particular IP address; commonly, Uniform Resource Indicators (URIs) with a user and host component serve this purpose. To help telephone numbers work similarly on the Internet, a number of efforts have specified mechanisms to manage and retrieve information about telephone numbers via network services.

For example, the ENUM ([RFC6116]) effort specified a public DNS profile for translating telephone numbers into URIs. Due to the difficulty of coordinating the public administration of telephone numbers in the DNS, this work transitioned to "infrastructure" ENUM ([RFC5067]), which assumed private DNS implementations, each of which could give a different answer to the same request to translate a telephone number depending on who asked, or other internal factors. The framework of the SPEERMINT working group ([RFC6406]), expanding on these requirements, differentiated the mapping of a telephone number to a target network (the "Look-up Function") from the mapping made by the originating network to the proper next-hop to reach such a target network (the "Location Routing Function"). To provision the data associated with telephone numbers, the DRINKS working group ([RFC6461]) designed systems for uploading back-end data to the services that would answer ENUM queries.

None of the preceding efforts, however, encompassed the entire lifecycle of a telephone number as an Internet identifier. They focused largely on service data, on how to "resolve" a telephone number to a location on the Internet, rather than on administrative questions of how numbers are acquired, how the entities associated with telephone numbers are authorized to provision data, and how what kinds of systems need to be in place to allow a diverse community of devices, applications and uses to manage numbers. Early considerations were moreover based on overlapping, but not entirely consistent, information models: intrinsic limitations in the DNS kept the queries and responses of ENUM relatively simple, whereas the DRINKS provisioning system considered a much richer syntax.

The need for solutions in this space is pressing, as many carriers worldwide contemplate migrating their entire PSTN infrastructure onto the Internet within the next decade. Further pressures come from emerging Internet communications providers who never invested in PSTN infrastructure in the first place, but want access to services related to telephone numbers. This includes devices, services, and applications on the Internet that make use of telephone numbers and need to distribute and manage numbering inventory: for example, an Internet-enabled PBX that might want to automate the process for allowing new connected phones to acquire numbers and provision contact information for their users. These different communities have diverse requirements. In some environments, there are performance constraints that would require a very lightweight binary protocol; in others, applications might prefer human-readable markup languages suitable for interfacing with existing APIs. The use cases associated with these functions are detailed in [I-D.peterson-modern-problems].

Therefore, this document proposes a reconsideration of telephone service and administration data on the Internet, based on an information model that allows records associated with telephone number to be created, modified and accessed through network interfaces. This document specifies no particular syntax or encoding for queries or responses, but instead describes an extensible information model for the semantics of provisioning and querying operations associated with a telephone number.

3. The Information Model

The fundamental building block of the TeRI model is the Record. A Record is created by an Authority who has authority over a particular telephone number or a set of numbers. There may be more than one Authority who is authorized to create Records for a particular telephone number, and a TeRI service may have multiple Records corresponding to a single telephone number, including potentially

Records associated with a range of numbers including a particular telephone number. Under various circumstances detailed in Section 4, participants in the numbering ecosystem may create, read, update, and modify Records.

Records contain Elements that hold data about the telephone number. Elements in this information model have a Name, which may optionally be associated with a Type and Value. Elements are grouped into Service Elements and Administrative Elements.

3.1. Record Elements

A Record is made up of Elements, which may be either Service Data Elements or Administrative Data Elements.

3.1.1. Identifier

Every Record has an Identifier, which is a globally unique identifier of the Record. The Identifier will typically be created at the same time as the Record itself, at a time when an assignment or delegation has occurred (as described in [I-D.peterson-modern-problems]).

3.1.2. Authority

Every Record contains an Authority element the source of the data: either the entity that provisioned the data with the Service, or the external source from which the Service collected the data. The Authority element ideally gives a logical identity of the source of the data. A public key value may also be designated by the Authority element.

3.1.3. Contact

Every Record has at least one Contact. The Contact contains administrative data about the assignee of the telephone number, though additional Contacts may contain information about delegates (as defined in [I-D.peterson-modern-problems]).

3.1.4. Subject

Every Record has a Subject. As the TeRI record concerns telephone numbers, the Subject of a Record is either a telephone number type or a telephone number range type.

3.1.5. Service

Records optionally have one or more Service entries. A Service may be of any Service Type, as given in Section 3.2.1.

3.1.5.1. Priority

Optionally, a Service may specify a weighted Priority associated with a Record. Priorities are between 0 and 1, with a value of 1 having the highest priority.

3.1.5.2. Expiration

Optionally, a Service may specify an absolute time at which a Record will no longer be valid, should a client or intermediary wish to cache a Record. In the absence of an Expiration element, Records may be cached for a maximum of twenty-four hours.

3.1.6. Signature

Optionally, a Record contains a Signature element. The Signature element contains a signature over the concatenation of the other elements given the Record. Signatures are provided by the Authority responsible for the Record.

[Syntax TBD]

3.2. Element Value Types

The remainder of a Record is made up of Elements. Elements types are specified in this section. Every Element Type has a Type Code. A Type Code is used as a short form for the Element in a Record.

3.2.1. Service Types

3.2.1.1. Telephone Number Type

The telephone number type conforms to the telephone number syntax given in [RFC3966] Section 3, in the ABNF for "telephone-subscriber."

Type Code: T

[TBD - need for subtying? E.164, Service Code, Short Code, Prefix, Nationally-Specific and Unknown.]

3.2.1.1.1. TN Range Type

The TN range type consists of a prefix of a telephone number (per [RFC3966] "telephone-subscriber"), and is semantically equivalent to all syntactically-valid telephone numbers below that prefix. For example, in the North American Numbering plan, the prefix 157143454 would be equivalent to all numbers ranging from 15714345400 to 15714345499.

[TBD - identify alternative ways of specifying ranges, potentially as separate element types]

Type Code: R

3.2.1.1.2. Domain Name Type

The domain name type conforms to the syntax of RFC1034 Section 3.5 and Section 2.1 of [RFC1123].

Type Code: D

3.2.1.1.3. Uniform Resource Indicator (URI) Type

The Uniform Resource Indicator (URI) type conforms to the syntax for URIs given in [RFC3986] (see Section 3).

Type Code: U

3.2.1.1.4. Internet Protocol (IP) Address Type

The IP Address type conforms to the ABNF syntax of either the IPv4address given in RFC3986 (Appendix A) or the IPv6reference of [RFC5954].

Type Code: I

3.2.1.1.5. Trunk Group Type

The trunk group type conforms to the "trunk-group-label" ABNF given in [RFC4904] (Section 5).

Type Code: G

3.2.1.1.6. Service Provider Identifier (SPID) Type

The SPID type consists of a four-digit number.

[TBD - introduce other elements for alternative SPID syntaxes]

Type Code: ?

3.2.2. Public Key Type

The Credential type consists of a public key [encoding TBD].

Type Code: C

3.2.3. Contact Type

The contact type follows the conventions of jCard [RFC7095].

Type Code: C

3.2.4. Expiry Type

The Expiry type is an absolute time conformant to the syntax of [RFC3339].

Type Code: E

3.2.5. Priority Type

The Priority type contains a number between 0 and 1, conforming to the specification of the "q" parameter of the Contact header field in [RFC3261].

Type Code: P

3.2.6. Record Identifier Type

The Record Identifier Type consists of a unique identifier for a record [format TBD].

Type Code: U

3.2.7. Signature

[Syntax TBD]

Type Code: S

3.2.8. Extension Type

This code is reserved for future use.

Type Code: X

4. Operations

In this section are detailed the three TeRI Operations: Acquisition, Management, and Retrieval Operations.

4.1. Common to All Operations

All Operations in the TeRI model consist of Requests and Responses. A Request from a TeRI client to a service may attempt to create, read, update, or delete TeRI Records. Requests may focus only on particular parts of a TeRI record. A Response gives the result of the Operation back to the client, which may indicate success or failure.

4.1.1. Requests

All TeRI Requests have a Source, a Subject, and optionally a set of Attributes which further specify the nature of the Request. Some Requests will know the Identifier of the Record they concern, and may convey that in an Attribute; others will query for all Records matching a given Subject.

4.1.1.1. Source

The Source is a required element in all Requests. In this specification, two categories of Sources are defined: Request Source and Request Intermediary. At least one of these Sources must be present in a Retrieval Request, and multiple Sources are permitted. Responses do not contain a Source.

Future specifications may extend the set of Source types.

4.1.1.1.1. Request Source

Every Request generated by a Client has a Request Source, which identifies the originator of the Request. This represents the logical identity of the user or service provider who first sent the Request, rather than the identity of any Intermediate entity. This field is provided in the Source to authenticate the poser of the Request, so that the Service can make any necessary authorization decisions as it formulates a Response.

In some service deployments, an Intermediary may wish to mask the Request's Source from a Service. The removal of the Request's Source by an intermediary is permitted by TeRI, but any Intermediary that removes the Request Source must provide a Request Intermediary for the Source element.

A Request Source element has a Type, which indicates how the logical identity of the originator of the Request has been represented. The Type field of the Request Source is extensible. Initial values include a domain name, a URI and a telephone number.

The Type element of the Request Source is followed by a Value, which contains the identity. The format of the identity is determined by the Type.

4.1.1.1.2. Request Intermediary

Optionally, Requests may contain one or more Request Intermediary elements in the Source. A Request Intermediary resides between the originator of the Request (the Client) and the Service, where it may aggregate queries, proxy them, transcode them, or provide any related relay function to assist the delivery of Requests to the Service.

The Request Intermediary element, like the Request Source, contains the logical identity of the service that relayed the Request. This field is provided in the Source for those deployments in which the Service makes an authorization decision based on the identity of the Intermediary rather than a Request Source.

A Request Intermediary element has a Type, which indicates how the logical identity of the Intermediary has been represented. The Type element of the Request Intermediary is extensible. Initial values include a domain name, an X.509 certificate subject, or a URI.

The Type of the Request Intermediary element is followed by a Value, which contains the identity. The format of the identity is determined by the Type.

4.1.1.2. Subject

All Requests have a Subject. The Subject identifies the resource that the Request concerns. Responses only contain a Subject if the Subject of the Response differs from that of the original Request, which may occur when (for example) the Subject contains a broad range, and the Service replies with a more narrow Subject. Future specifications, including Profiles, may define alternative Subject elements.

4.1.1.2.1. Attributes

TeRI Attributes consist of a Name with an optional Type and an Optional Value. Most Attributes are specific to the Operation.

4.1.2. Responses

All TeRI responses consist of a Response Code and optionally a set of Attributes which convey further information about the Operation. Most Attributes are specific to the Operation.

4.1.2.1. Response Code

All Responses contain a Response Code.

Response Codes defined by this document include: Success, Subject Does Not Exist, Subject Conflict, No Suitable Records Exist for Subject, Subject Syntax Error, Unknown Attribute, Unauthorized Source, Route Source Topology Unavailable.

[TBD]

4.2. The Acquisition Operation

An Acquisition Request has a Source and a Subject, and may have one or more Attributes. An Acquisition Response has a Response Code, and will contain one Record if it is successful.

The Subject of an Acquisition Request always specifies a Telephone Number Type or a Telephone Number Range Type. If the Subject contains a particular telephone number, then the Acquisition Request is a Request to acquire that particular telephone number. If it is a range, the Acquisition Request should be considered to be for the entire range, but Attributes of the Request may limit the scope of the resources requested. The Service will determine whether or not the Client is authorized to acquire the resources in question based on the Source of the Acquisition Request.

The Response to an Acquisition Request will contain a Success Response Code if the resource can be allocated. The Subject of a Success Response will always contain the Telephone Number Type or Telephone Number Range that has been allocated. A successful Acquisition Response must contain a Record with a Identifier Element; that Record may also contain a Public Key attribute. By default, this Record will contain only Administrative Elements, without Service Elements. If a requested telephone number (or range) is already allocated, or a telephone number in the specified range is not available, then a Subject Conflict Response Code is returned.

4.3. The Management Operation

A Management Request comprises a Source, a Subject, and one or more Records; it also may contain one or more Attributes. A Management Request contains a Response Code, and optionally may contain a Record.

The Subject of a Management Request always specifies a Telephone Number Type or a Telephone Number Range Type. If the Subject contains a particular telephone number, then the Acquisition Request is a Request to acquire that particular telephone number. If it is a range, the Acquisition Request should be considered to be for the entire range.

A Management Request contains at least one Record; it may contain multiple Records. Each Record in the Management Request must contain a Record Identifier Element which designates the Record that the Client is requesting that the Service replace with the Record included in the Management Request. The Service will determine whether or not the Client is authorized to modify the Record in question via the Source of the Management Request.

4.4. The Retrieval Operation

Every Retrieval Request comprises a Source and a Subject, and may have one or more Attributes. A Retrieval Response has a Response Code, optionally one or more Records, and optionally a Subject, if the Subject differs from that of the Request.

Retrieval Requests optionally contain Attributes; a Request with no specified Attributes requests that the Service return any Attributes associated with the Subject. In a Request, the presence of one or more Attributes limits the scope of the Request to Records about the Subject containing those Attributes. Typically an Attribute will specify a Service or Service Type that the Client seeks Records for.

Retrieval Responses contain one or more Records. At least one Record will always be present in a successful Response.

4.5. Common Attributes

Attributes are broadly divided between Service Attributes and Administrative Attributes. Service Attributes provide information required to route communications, including URIs. The format of the elements contained in the Attributes is given in Section 3.2.

4.5.1. Administrative Attributes

Administrative Attributes defined by this document include: CNAM (Type Display Name), SPID (Type SPID), dialplan (Type ?) [TBD]

4.5.2. Service Attributes

Service Attributes defined by this document include: voip (Type URI), sms (Type URI) [TBD]

4.5.2.1. Route Source

Optionally, Retrieval Requests may contain a Route Source Attribute which identifies a reference point in the network from which any Service Attributes in the response should be calculated. It therefore always designates a network element, though depending on the circumstances, it may be an endpoint, a gateway, a border device, or any other agent that makes forwarding decisions for telephone calls and related services.

A Route Source element has a Type, which indicates how the network element has been represented. The Type field of the Request Source is extensible. Initial values include a domain name, an IP address or a trunk group.

The Type of the Route Source element is followed by a Value, which designates the network element. The format of the identity is determined by the Type.

5. Implementing Operations

This framework specifies an abstract Request/Response protocol that enables a Client to send Requests to a Service about telephone numbers or related telephone services. Requests may pass through one or more Intermediaries on their way from a Client to a Service; for example, through aggregators or service bureaus. A Client establishes the Subject of a Request, and optionally includes one or more Attributes to focus the scope of the Request. When a Service receives a Request, it performs any necessary authorization and policy decisions based on the Source. If policy permits, the Service generates a Response, which will consist of a Response Code and one or more Records associated with the Subject. The Service then sends the Response through the same path that the Request followed; transactional identifiers set by the Client and Service correlate the Request to the Response and assist any intermediary routing.

5.1. Transport Independence

The information model provided for Requests and Responses in this framework is independent of any underlying transport or encoding. Future specifications will define Bindings that specify particular transports and Encodings for Requests and Responses. In some deployment environments, for example, a binary encoding and lightweight transport might be more appropriate than the use of a web protocol. This specification provides a template of requirements that must be addressed by any encoding scheme.

It is a design goal of this work that the semantics of Requests and Responses survive interworking through translations from one encoding to another; for example, when an Intermediary receives a binary Request from a Client, it should be able to transcode it to an XML format to send to a Service without discarding any of the original semantics.

5.2. Bindings

A TeRI Binding is an underlying protocol that carries Requests and Responses. Future specifications may define Bindings in accordance with the procedures in the IANA Considerations sections of this document.

By underlying protocol, this specification means both transport-layer protocols as well as any application-layer protocols that the Binding requires. Thus an example Binding might specify a combination of TCP, TLS, HTTP and SOAP as the underlying transport for TeRI. Alternatively, it might only specify a very lightweight underlying protocol like UDP. A Binding may be specific to a particular Encoding, or it may be independent of any Encoding.

Bindings must specify whether they are continuous, transactional or non-transactional. A continuous Binding creates a persistent connection between two TeRI entities over which many, potentially unrelated, Requests and Responses might flow. Many Bindings defined for use between an Intermediary and a Service will have this property, as Intermediaries may aggregate on behalf of many Clients, and opening a separate transport-layer connection for each new Request would be inefficient. A transactional Binding creates a temporary connection between two TeRI entities for the purpose of fulfilling a single Request; any Responses to the Request will use the same connection to return to the sender of the Request. Finally, a non-transactional Binding does not rely on any sort of connection semantics: the senders of Requests and Responses will always initiate a new instance of the Binding to send a message.

This document makes no provision for discovering the Bindings supported by a TeRI Client, Intermediary or Service. Intermediaries may transcode between Bindings if necessary when acting to connect a Client and a Service, especially if the Client and Service support no Bindings in common.

A Binding specification must enumerate all categories of metadata required to establish a connection using a Binding. For some Bindings, this might comprise solely an IP address and a port; for other Bindings, this might instead require higher-layer application identifiers like a URI. This metadata includes any identifiers necessary for correlating Requests to Responses in a continuous or non-transactional Binding; any Encoding making use of these Bindings must specify how it carries those elements.

Bindings must also describe the security services they make available. Bindings must have a means of providing mutual authentication, integrity and confidentiality between Clients, Intermediaries and Services. If a Binding supports TLS, for example, these features can be provided by using TLS in an appropriate deployment environment.

5.3. Encodings

A TeRI Encoding specifies how the Request and Response are constructed syntactically. An Encoding may be specific to a particular Binding, or it may be specified independently of any Binding.

An Encoding may define an object format; for example, an XML or JSON object, described with any appropriate schemas, or an ABNF description. An Encoding might alternatively specify a mapping of the semantic elements of Requests and Responses on to the existing fields of headers of a protocol, especially when that protocol has been defined as an underlying protocol Binding. Encodings must also define whether or not they provide a bundling feature that allows multiple Requests to be carried within particular objects or mappings.

Every Encoding must specify how each semantic Element Type of a Request and Response will be represented. For all baseline TeRI Attributes and Element Types, the Encoding specifies whether values will be text or binary, how they will be encoded. Many baseline Element Types (such as telephone numbers) can appear in different places in a TeRI message; Encodings need only specify these common element types once. Due to the extensibility of TeRI, however, future specifications might define Element Types that an Encoding

does not address. Profiles using those extensions and Encodings must explain their interaction.

Encodings must also describe the security services they make available. In particular, encodings must describe a means of providing authentication of the Sources and Authorities of Requests and Responses respectively, as well as an integrity check over critical elements including the Subject of Requests and the Record of Responses.

[TBD - we may define more about the computation of this signature, including canonicalization of elements, in this framework, and make it a requirement for encodings to support this mechanism]

5.4. Profiles

For particular deployment environments, only one Binding, Encoding and set of Attributes or other extended elements may be meaningful. Future specifications may therefore define TeRI Profiles, which describe a particular deployment environment and the Binding, Encoding and set of Attributes or elements it requires.

Profiles may be extensible, but any Attributes or elements required to negotiate support for extensions must be defined within the Profile.

6. Security Considerations

The framework of this document differs from previous efforts to manage telephone numbers on the Internet largely by offering a much richer set of security services. In particular, it requires that three entities be capable of authenticating themselves to one another at the layer of a binding: Clients, Intermediaries and Services. It furthermore requires object security at the encoding layer so that Sources and Authorities can sign data in order to authenticate Requests and Responses that may pass through Intermediaries, and moreover so that Authorities can prove to Clients that their Records are authoritative even when the Authority does not operate the Service. The requirements that bindings and encodings must satisfy to meet these security needs are specified in Section 5.1.

[TBD - more]

7. IANA Considerations

This specification defines several registries: A registry of Elements, a registry of Element Types, a registry of Attributes, and a registry of Response Codes.

This document creates a registry of Elements for use with this framework. This registry is extensible, with an IANA Registration policy of Specification Required. Any new Element registered must supply the name of the Element, the name of the parent Element in the information model, and a code point. [TBD]

This specification pre-provisions the Element Types registry with the entries given in Section 6. These elements are indexed by their Type Code. This registry is extensible, with an IANA Registration policy of Specification Required. Any new Element Type registered must supply the name of the Element Type, the name of the parent element in the information model, and a Type Code.

This specification creates an Attribute registry which is indexed by Attribute names. This registry is extensible, with an IANA Registration policy of Specification Required. Any new element registered must supply the name of Attribute, and list all Element Types that may be associated with Values of the Attribute.

This document furthermore creates a registry of Response Codes. This registry is pre-provisioned with the values given in Section 5.5. [TBD]

8. Acknowledgements

The authors would like to thank Paul Kyzviat and Dale Worley for their input into this specification.

9. Informative References

- [I-D.peterson-modern-problems] Peterson, J. and T. McGarry, "Modern Problem Statement, Use Cases, and Framework", draft-peterson-modern-problems-01 (work in progress), July 2015.
- [RFC1123] Braden, R., Ed., "Requirements for Internet Hosts - Application and Support", STD 3, RFC 1123, DOI 10.17487/RFC1123, October 1989, <<http://www.rfc-editor.org/info/rfc1123>>.
- [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>>.
- [RFC3324] Watson, M., "Short Term Requirements for Network Asserted Identity", RFC 3324, DOI 10.17487/RFC3324, November 2002, <<http://www.rfc-editor.org/info/rfc3324>>.
- [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>>.
- [RFC3339] Klyne, G. and C. Newman, "Date and Time on the Internet: Timestamps", RFC 3339, DOI 10.17487/RFC3339, July 2002, <<http://www.rfc-editor.org/info/rfc3339>>.
- [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>>.
- [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>>.
- [RFC4904] Gurbani, V. and C. Jennings, "Representing Trunk Groups in tel/sip Uniform Resource Identifiers (URIs)", RFC 4904, DOI 10.17487/RFC4904, June 2007, <<http://www.rfc-editor.org/info/rfc4904>>.
- [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>>.
- [RFC5039] Rosenberg, J. and C. Jennings, "The Session Initiation Protocol (SIP) and Spam", RFC 5039, DOI 10.17487/RFC5039, January 2008, <<http://www.rfc-editor.org/info/rfc5039>>.

- [RFC5067] Lind, S. and P. Pfautz, "Infrastructure ENUM Requirements", RFC 5067, DOI 10.17487/RFC5067, November 2007, <<http://www.rfc-editor.org/info/rfc5067>>.
- [RFC5727] Peterson, J., Jennings, C., and R. Sparks, "Change Process for the Session Initiation Protocol (SIP) and the Real-time Applications and Infrastructure Area", BCP 67, RFC 5727, DOI 10.17487/RFC5727, March 2010, <<http://www.rfc-editor.org/info/rfc5727>>.
- [RFC5954] Gurbani, V., Ed., Carpenter, B., Ed., and B. Tate, Ed., "Essential Correction for IPv6 ABNF and URI Comparison in RFC 3261", RFC 5954, DOI 10.17487/RFC5954, August 2010, <<http://www.rfc-editor.org/info/rfc5954>>.
- [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, <<http://www.rfc-editor.org/info/rfc6116>>.
- [RFC6406] Malas, D., Ed. and J. Livingood, Ed., "Session PEERing for Multimedia INTERconnect (SPEERMINT) Architecture", RFC 6406, DOI 10.17487/RFC6406, November 2011, <<http://www.rfc-editor.org/info/rfc6406>>.
- [RFC6461] Channabasappa, S., Ed., "Data for Reachability of Inter-/Intra-Network SIP (DRINKS) Use Cases and Protocol Requirements", RFC 6461, DOI 10.17487/RFC6461, January 2012, <<http://www.rfc-editor.org/info/rfc6461>>.
- [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>>.
- [RFC6950] Peterson, J., Kolkman, O., Tschofenig, H., and B. Aboba, "Architectural Considerations on Application Features in the DNS", RFC 6950, DOI 10.17487/RFC6950, October 2013, <<http://www.rfc-editor.org/info/rfc6950>>.
- [RFC7095] Kewisch, P., "jCard: The JSON Format for vCard", RFC 7095, DOI 10.17487/RFC7095, January 2014, <<http://www.rfc-editor.org/info/rfc7095>>.

[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>>.

Author's Address

Jon Peterson
Neustar, Inc.

Email: jon.peterson@neustar.biz

Internet Engineering Task Force
Internet-Draft
Intended status: Standards Track
Expires: April 21, 2016

H. Bellur
C. Wendt, Ed.
Comcast
October 19, 2015

Distributed Registry Protocol
draft-wendt-modern-drip-00

Abstract

This document describes a protocol for allowing a distributed set of nodes to synchronize a set of information in real-time with minimal amount of delay. This is useful for registry types of information like identity and telephone numbers with associated routing and ownership information and could be extended to support other distributed real-time information updates as well.

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 April 21, 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
1.1. Terminology	2
2. DRiP Overview	3
3. Distributed MESH Architecture	3
4. DRiP procedures	4
4.1. Distributed Registry Rules	4
4.2. Node State	5
4.2.1. API - POST /node/:nodeid/active	5
4.2.2. API - POST /node/:nodeid/inactive	5
4.2.3. API - GET /state	5
4.3. Custom HTTP header fields	6
4.4. Key-Value Data Propagation Rules	8
4.5. Key-Value Data Update	8
4.5.1. Voting Phase	9
4.5.1.1. API - POST /voting	10
4.5.1.2. POST /votingphase/node/:nodeid/response/:response	11
4.5.2. Commit Phase	11
4.5.2.1. API - POST /commit	12
4.6. Node Sync Operation	13
4.6.1. API - PUT /sync/node/:nodeid	13
4.7. Heartbeat	14
4.8. Key-Value Data Update Entitlement Verification	14
5. Security Considerations	14
5.1. HTTPS	14
5.2. Authentication	14
6. References	14
Authors' Addresses	15

1. Introduction

This document describes the Distributed Registry Protocol (DRiP). DRiP defines a set of peer protocols for how an arbitrary number of nodes arranged in a distributed mesh architecture can be used to synchronize data in real-time across a network.

1.1. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

Initiator Node:

A node that initiates data propagation.

Receiver Node:

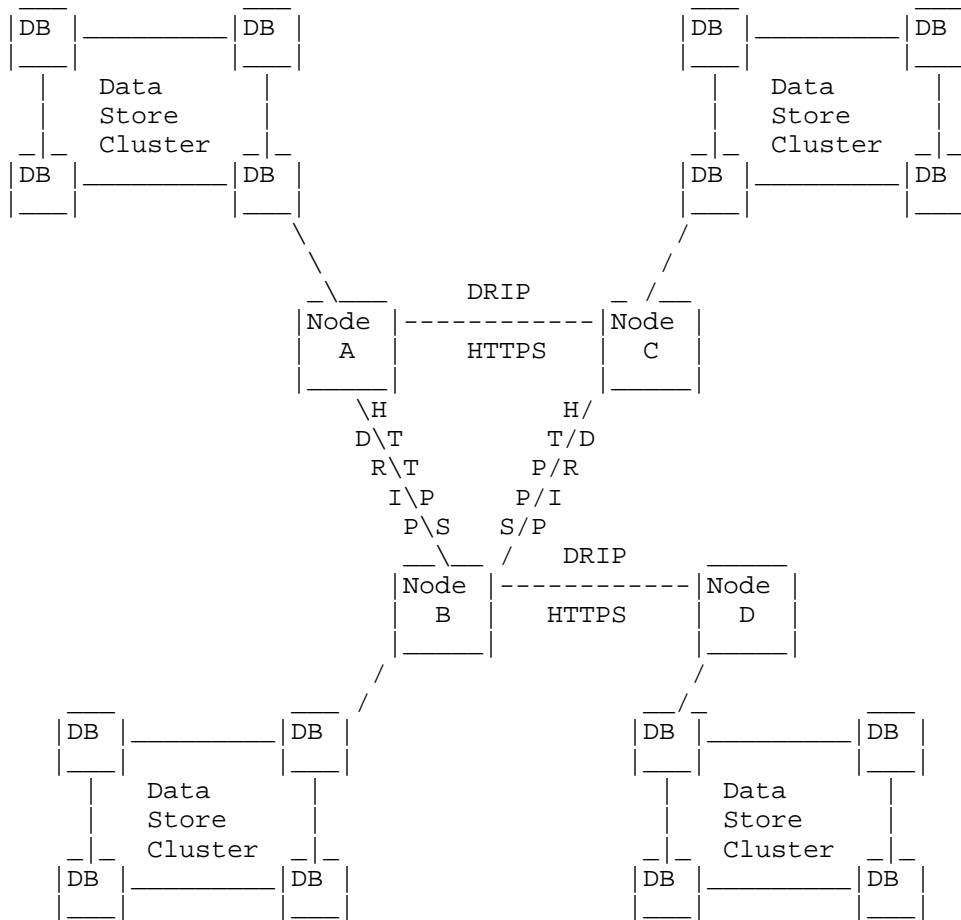
A node that forwards the propagated key-value data.

2. DRiP Overview

DRiP uses a mix of a gossip protocol with update counters for distribution of key-value data with the addition of a voting system to avoid race conditions on writing of key-value data.

3. Distributed MESH Architecture

The DRiP architecture is based on a peer-to-peer communication model where a given node associated with a data store is not necessarily aware of the total number of nodes in the entire network. Minimally, every node should be reachable by at least one multi-node path from every other node. Each node in the DRiP network maintains a list of peer nodes from which it receives and transmits updates. Information is propagated by forwarding to its peer nodes until the information received by a node has already been received.



Distributed Mesh Architecture

4. DRiP procedures

4.1. Distributed Registry Rules

All nodes in the distributed mesh MUST agree upon a specific key-value data model. The choice of data store is implementation specific.

All nodes MUST be configured with at least one peer node before propagation.

A node MUST ignore any updates or commands it receives from other nodes that are not configured as peer nodes.

All nodes MUST send a periodic heartbeat or keep-alive message via HTTPS to the respective peer nodes. If a heartbeat is not received the peer node is removed from the list of active peer nodes.

4.2. Node State

The peer node should maintain a state that defines whether it is active, inactive, or synchronizing key-value data with a peer node.

The node should proactively tell it's peer nodes its state by sending the following POST messages. The GET query is available for nodes to query the state of peer nodes.

4.2.1. API - POST /node/:nodeid/active

Request:

POST /node/:nodeid/active

Description:

TBD

Example:

TBD

4.2.2. API - POST /node/:nodeid/inactive

Request:

GET /state

Description:

A node should query the state of its peer node before it initiates a sync operation. This request responds with either "active" or "sync" or no response, if in "inactive" state.

Example:

TBD

4.2.3. API - GET /state

Request:

POST /node/:nodeid/active

Description:

TBD

Example:

TBD

4.3. Custom HTTP header fields

Custom HTTP header fields will be used to carry node specific information.

Field Name	Description
DRiP-Node-ID	Each node in the mesh MUST have a unique identifier. An Initiator node MUST set its own node ID as the field value. A Receiver Node MUST NOT change the DRiP-Node-ID field value as it forward the HTTPS request to its peer nodes.

Example:

DRiP-Node-ID: xyz

Field Name	Description
DRiP-Node-Counter	Every node maintains a count of the number of times it initiates key-value data propagation. This counter MUST be an unsigned type, typically, a 64 bit integer. The Initiator node MUST set this count as the field value. A Receiver Node MUST NOT change the DRiP-Node-Counter field value as it forward the HTTPS request to its peer nodes.

Example:

DRiP-Node-Counter: 123

Field Name	Description
DRiP-Node-Counter-reset	A node can reset the count (to zero) of the number of times it initiates key-value data propagation. If the counter value is reset, prior to initiating data propagation, then this field value MUST be set to true. Otherwise, it MUST be set to false, at all times. A typical use case to reset the counter value is when the counter (of unsigned type) value wraps around. The Initiator node MUST set this field value to either true or false. A Receiver Node MUST NOT change the DRiP-Node-Counter-reset field value as it forward the HTTPS request to its peer nodes.

Example:

DRiP-Node-Counter-reset: false

Field Name	Description
DRiP-Transaction-Type	The Initiator node MUST set this field value to be either "update" or "sync". A Receiver Node MUST NOT change the DRiP-Transaction-Type field value as it forward the HTTPS request to its peer nodes.

Example:

DRiP-Transaction-Type: update

Field Name	Description
DRiP-Sync-Complete	For sync transaction type, the Initiator node MUST set this field value to be true, if synchronization is complete. Otherwise, this field value MUST be set to false.

Example:

DRiP-Sync-Complete: false

4.4. Key-Value Data Propagation Rules

A node propagates key-value data to all its peer nodes except the node from which it received data. For example, in Figure 1, when node B receives key-value data from node A, it will propagate the data received to nodes C and D but not back to node A.

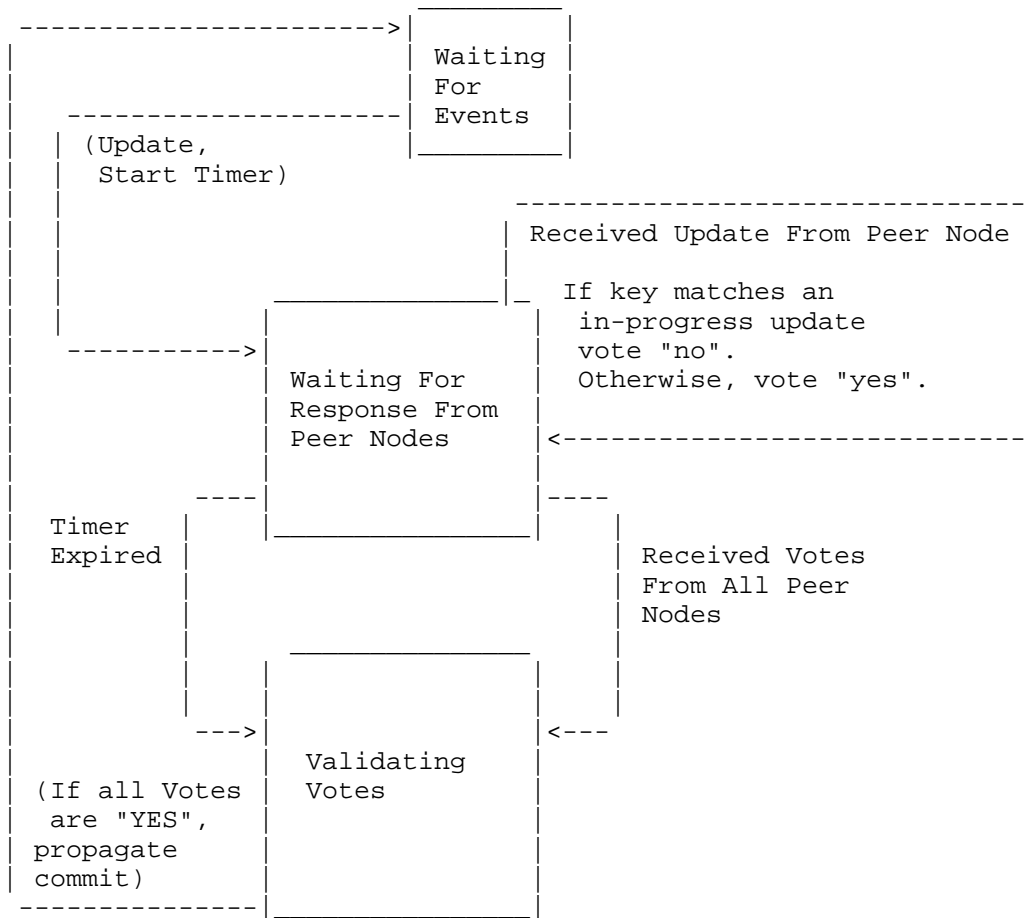
For each transaction type (Update or Sync), the following set of actions MUST take place when a node receives a HTTPS request with propagated key-value data:

- o If DRiP-Node-ID field value (in the HTTP header) contains Initiator node ID that has never been seen, both DRiP-Node-ID and DRiP-Node-Counter field values MUST be stored for future reference and the key-value data is propagated to all peer nodes.
- o If DRiP-Node-ID field value (in the HTTP header) matches with a stored node ID and DRiP-Node-Counter-reset field value is false.
 - * The received key-value data MUST be propagated to the peer nodes if DRiP-Node-Counter field value is greater than the saved counter value. The DRiP-Node-Counter field value MUST be saved as the new counter for the stored node ID.
 - * If DRiP-Node-Counter field value is less than or equal to saved counter value, then the key-value data has already been received and MUST NOT be propagated to peer nodes. This ensures that propagation stops when all nodes have received the key-value data from the Initiator node.
- o If DRiP-Node-ID field value matches with a stored node ID and DRiP-Node-Counter-reset field value is true:
 - * The received key-value data MUST be propagated to the peer nodes. The DRiP-Node-Counter field value MUST be saved as the new counter for the stored node ID.

4.5. Key-Value Data Update

When an Initiator node has new data it wants to propagate to the distributed mesh, it initiates an Update. The Update consists of a two-phase commit (2PC) procedure in order to guarantee there are no race conditions for updating the same key's data, as well as for any error conditions in the distributed mesh that would cause the update to not complete for all nodes in the network.

The two phases are called the "voting" phase and the "commit" phase.



Update State Diagram

4.5.1. Voting Phase

The voting phase is the phase where all nodes are queried to "vote" whether they are aware of any potential conflict that would cause the transaction not to complete.

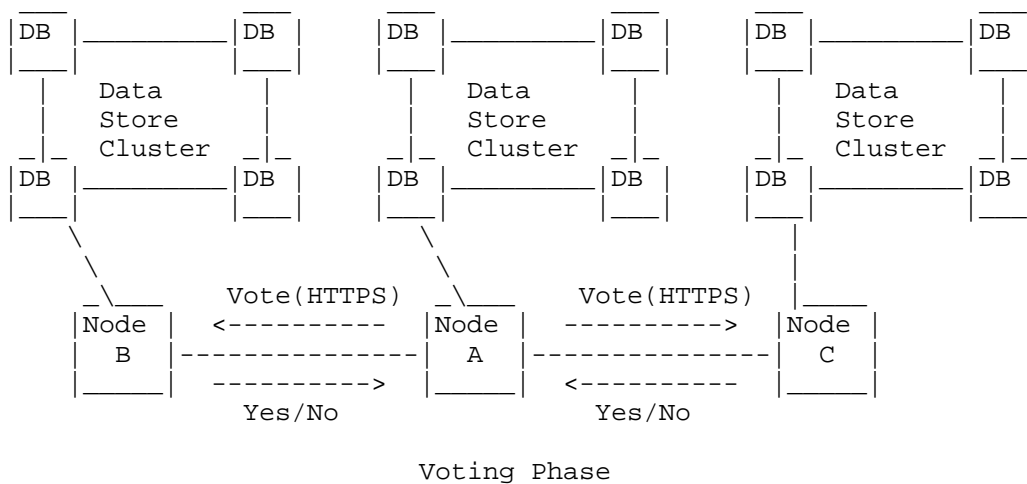
The Initiator node MUST set a timeout period to get response from its peer nodes.

The peer nodes known to the initiator node will continue propagate the information to their peer nodes and so on. However, these peer nodes beyond the initiator node will no longer need to keep track of the time interval for responses. A node will stop continuing to

propagate information when it determines it has received the same information again. This can be determined by keeping track of the counter and originating node id.

If all peer nodes vote "yes", then the second phase or commit phase in the local node is initiated. If any node in the distributed mesh votes "no" or if the timeout period expires and all peer nodes have not responded, then the commit of the information MUST NOT be completed. No action is taken for responses received after the timeout period.

Note: The voting procedure is intentionally split into two separate full HTTP transactions for reliability.



4.5.1.1. API - POST /voting

Request:

POST /voting

Description:

A post from either Initiator node or subsequent peer nodes to request a vote of "yes" or "no" whether the key-value data could be committed without error or conflict.

Example (using cURL)

Request

```
$ curl -i -H "Content-Type: application/json" -H "DRiP-Node-ID:
nodeA" -H "DRiP-Node-Counter: 1234" -H
"DRiP-Node-Counter-reset: false" -X POST -d '{<key-value
data>}' https://nodebregistry.com/voting
```

Response

```
HTTP/1.1 200 OK
```

4.5.1.2. POST /votingphase/node/:nodeid/response/:response

Request:

```
POST /voting/peernode/:nodeid/response/:response
```

Description:

A POST from peer node back to node with response of vote.

Example (using cURL)

Request

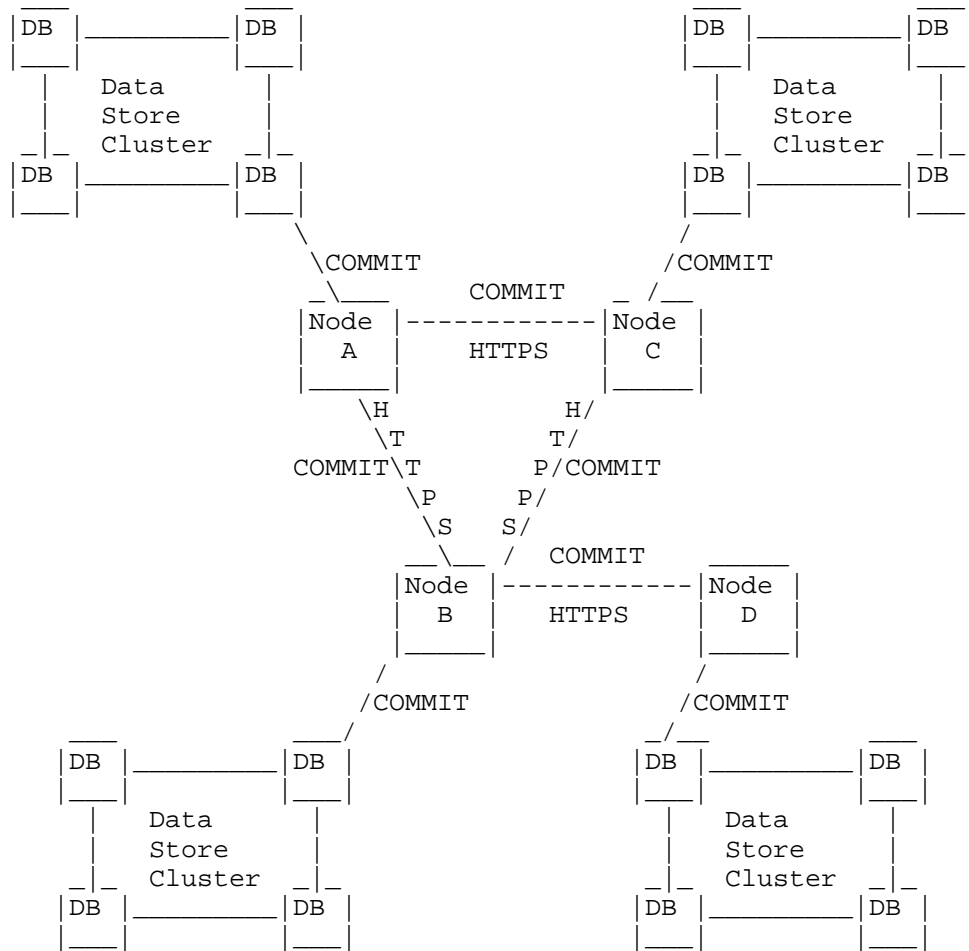
```
$ curl -i -X POST http://nodearegistry.com/node/nodeA/response/yes
```

Response

```
HTTP/1.1 200 OK
```

4.5.2. Commit Phase

The Initiator node, that originated the gossip, upon receiving a successful aggregated "yes" vote from all the peer nodes should start the commit phase. This node MUST commit the data to its data store. Subsequently, this information is propagated to all the nodes so that each node in the mesh will commit the same information in their respective data stores.



Commit Phase

4.5.2.1. API - POST /commit

Request:

POST /commit

Description:

A commit message is sent from Initiator or subsequent peer nodes to signal the Receiver node to commit the data to its data store.

Example (using cURL)

Request

```
$ curl -i -H "Content-Type: application/json" -H "DRiP-Node-ID:
nodeA" -H "DRiP-Node-Counter: 1234" -H
"DRiP-Node-Counter-reset: false" -X POST -d
'<key-value data>' https://nodebregistry.com/commit
```

Response

```
HTTP/1.1 200 OK
```

4.6. Node Sync Operation

A node, either newly added to the distributed mesh or put back into service after being inactive, will get the state of a peer node to determine if it is in "active" state. If so, the node can immediately initiate a Sync transaction. The peer node MUST start propagating a comprehensive and complete set of key-value data from its data store.

The two phase commit does NOT apply here as the contents of the initiating node's data store is either outdated or empty. During this phase (HTTPS requests received will have DRiP-Sync-Complete field value set to false), this node SHOULD NOT become an Initiator node to provision data. While this transaction is going on, this node MUST vote "yes" to all real-time updates. The commits corresponding to the Updates should also be completed and reflected in the data store.

4.6.1. API - PUT /sync/node/:nodeid

Request:

```
PUT /sync/node/:nodeid
```

Description:

API call for initiating a full registry synchronization from node to peer-node.

Example (using cURL)

Request

```
$ TBD
```

Response

HTTP/1.1 200 OK

4.7. Heartbeat

TBD - definition of heartbeat

TBD - determination of the need to sync after heartbeat fails

4.8. Key-Value Data Update Entitlement Verification

When a node owner would like to create or modify particular key-value data, generally in the context of a registry, there MAY be a verification procedure that key-value data write or modification can be performed. This could include validating whether key-value data is entitled to be written, modified or subsequently propagated based on application policy. For example, identity or telephone number ownership or porting. The exact mechanics of this are out of scope of this document and are generally application specific.

5. Security Considerations

5.1. HTTPS

All nodes MUST perform HTTP transactions using TLS as defined in [RFC7230].

5.2. Authentication

Secure authentication of node to node communication is beyond the scope of this document, however best practices in terms of protecting the node API interface should be followed.

6. References

[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>>.

[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, <<http://www.rfc-editor.org/info/rfc7230>>.

Authors' Addresses

Harsha Bellur
Comcast
One Comcast Center
Philadelphia, PA 19103
USA

Email: Harsha_Bellur@comcast.com

Chris Wendt (editor)
Comcast
One Comcast Center
Philadelphia, PA 19103
USA

Email: chris-ietf@chriswendt.net