Network Working Group
Internet-Draft
Intended status: Informational
Expires: January 8, 2017

J. Peterson
T. McGarry
NeuStar, Inc.

July 7, 2016

Modern Problem Statement, Use Cases, and Framework
draft-ietf-modern-problem-framework-01.txt

Abstract

The functions of the public switched telephone network (PSTN) are rapidly migrating to the Internet. This is generating new requirements for many traditional elements of 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 use cases 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 8, 2017.

Table of Contents

1. Problem Statement 3
2. Definitions 4
2.1. Actors 5
2.2. Data Types 7
2.3. Data Management Architectures 8
3. Framework 9
4. Use Cases 10
4.1. Acquisition 11
4.1.1. CSP Acquires TNs from Registrar 11
4.1.2. User Acquires TNs from CSP 12
4.1.3. CSP Delegates TNs to Another CSP 12
4.1.4. User Acquires TNs from a Delegate 13
4.1.5. User Acquires Numbers from Registrar 13
4.2. Management 13
4.2.1. Management of Administrative Data 13
4.2.1.1. CSP to Registrar 14
4.2.1.2. User to CSP 14
4.2.1.3. User to Registrar 15
4.2.2. Management of Service Data 15
4.2.2.1. CSP to other CSPs 15
4.2.2.2. User to CSP 16
4.2.3. Managing Change 16
4.2.3.1. Changing the CSP for an Existing Communications Service 16
4.2.3.2. Terminating a Service 16
4.3. Retrieval 17
4.3.1. Retrieval of Public Data 17
4.3.2. Retrieval of Semi-restricted Administrative Data . . 18
4.3.3. Retrieval of Semi-restricted Service Data 18
4.3.4. Retrieval of Restricted Data 18
5. Acknowledgments 19
6. IANA Considerations 19
7. Security Considerations 19
8. Informative References 20
Authors' Addresses 22

1. Problem Statement

The challenges of utilizing telephone numbers (TNs) on the Internet have been known for some time. Internet telephony provided the first 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]. The resulting database was designed to function in a manner similar to the systems that route calls in the PSTN. 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, e164.arpa, proved to be politically and practically challenging, and less centralized models have thus flourished. Subsequently, the DRINKS [4] framework showed ways that authorities might provision information about TNs at an ENUM service or similar Internet-based directory. These technologies have also generally tried to preserve the features and architecture familiar to the PSTN numbering environment.

Over time, Internet telephony has encompassed functions that differ substantially from traditional PSTN routing and management, especially as non-traditional providers have begun to utilize numbering resources. An increasing number of enterprises, over-the-top Voice over IP providers, text messaging services, and related non-carrier services have become heavy users of telephone numbers. An enterprise, for example, could deploy an IP PBX that receives a block of telephone numbers from a carrier and then in turn distribute those numbers to new IP telephones when they associate with the PBX. Internet services offer users portals where they can allocate new telephone numbers on the fly, assign multiple "alias" telephone numbers to a single line service, implement various mobility or find-me-follow-me applications, and so on. Peer-to-peer telephone networks have encouraged experiments with distributed databases for telephone number routing and even allocation.

This dynamic control over telephone numbers has few precedents in the traditional PSTN outside of number portability. Number portability has been implemented in many countries, and the capability of a user to choose and change their service provider while retaining their TN is widely implemented now. However, TN administration processes rooted in PSTN technology and policies dictate that this be an exception process fraught with problems and delays. Originally, processes were built to associate a specific TN to a specific service

provider and never change it. With number portability, the industry had to build new infrastructure, new administrative functions and processes to change the association of the TN from one service provider to another. Thanks to the increasing sophistication of consumer mobile devices as Internet endpoints as well as telephones, users now associate TNs with many Internet applications other than telephony. This has generated new interest in models similar to those in place for administering freephone services in the United States, where a user purchases a number through a sort of number registrar and controls its administration (such as routing) on their own, typically using Internet services to directly make changes to the service associated with telephone numbers.

Most TNs today are assigned to specific geographies, at both an international level and within national numbering plans. Numbering practices today are tightly coupled with the manner that service providers interconnect, as well as how TNs 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 increasing 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 TNs could depend little on geography. Adapting TNs to the Internet requires more security, richer datasets and more complex query and response capabilities than previous efforts have provided.

This document will create a common understanding of the problem statement related to allocating, managing, and resolving TNs in an IP environment. It outlines a framework and lists motivating use cases for creating IP-based mechanisms for TNs. It is important to acknowledge at the outset that there are various evolving international and national policies and processes related to TNs, and any solutions need to be flexible enough to account for variations in policy and requirements.

2. Definitions

This section provides definitions for actors, data types and data management architectures as they are discussed in this document. Different numbering spaces may instantiate these roles and concepts differently: practices that apply to non-geographic freephone numbers, for example, may not apply to geographic numbers, and practices that exist under one Numbering Authority may not be permitted under another. The purpose of this framework is to

identify the characteristics of protocol tools that will satisfy the diverse requirements for telephone number acquisition, management, and retrieval on the Internet.

2.1. Actors

The following roles of actors are defined in this document:

Numbering Authority: A regulatory body within a country that manages that country's TNs. The Numbering Authority decides national numbering policy for the nation, region, or other domain for which it has authority, including what TNs can be allocated, and which are reserved.

Registry: An entity that administers the allocation of TNs based on a Numbering Authority's policies. Numbering authorities can act as the Registries themselves, or they can outsource the function to other entities. There are two subtypes of Registries: an Authoritative Registry and a Distributed Registry. The general term Registry in this document refers to both kinds of Registries.

Authoritative Registry: An authoritative Registry is a single entity with sole responsibility for specific numbering resources.

Distributed Registry: Distributed Registries are multiple Registries responsible for the same numbering resources.

Registrar: An entity that distributes the telephone numbers administered by a Registry; typically, there are many Registrars that can distribute numbers from a single Registry, through Registrars may serve multiple Registries as well. A Registrar has business relationships with its assignees and collects administrative information from them.

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

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 an individual, company or other entity may act in one or more of the roles above; for example, a company may be a CSP and also a Registrar. Although Numbering Authorities are listed as actors, they are unlikely to actually participate in the protocol flows themselves, though in some situations a Numbering Authority and Registry may be the same administrative entity.

All actors that are recipients of numbering resources, be they a CSP, Service Enabler, or User, can also be said to have a relationship to a Registry of either an assignee or delegate:

Assignee: An actor that is assigned a TN directly by a Registrar; an assignee always has a direct relationship with a Registrar.

Delegate: An actor that is delegated a TN from an assignee or another delegate, who does not necessarily have a direct relationship with a Registrar. Delegates may delegate one or more of their TN assignment(s) to one or more further downstream subdelegates.

As an example, consider a case where a Numbering Authority also acts as a Registry, and it issues 10,000 blocks of TNs to CSPs, which in this case also act as Registrars. CSP/Registrars would then be responsible for distributing numbering resources to Users and other CSPs. In this case, an enterprise deploying IP PBXs also acts as a CSP, and it acquires number blocks for its enterprise seats in chunks of 100 from a CSP acting as a Registrar with whom the enterprise has a business relationship. The enterprise is in this case the assignee, as it receives numbering resources directly from a Registrar. As it doles out individual numbers to its Users, the enterprise delegates its own numbering resources to those Users and their communications endpoints. The overall ecosystem might look as follows.

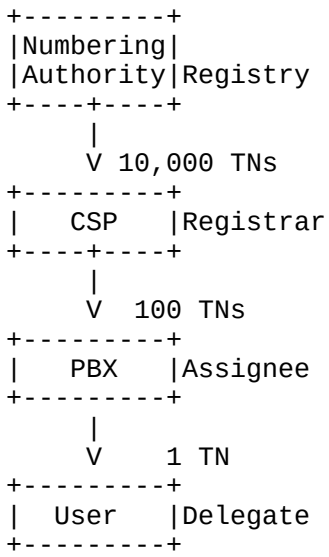


Figure 1: Chain of Number Assignment

2.2. Data Types

The following data types are defined in this document:

Administrative Data: assignment data related to the TN and the relevant actors; it includes TN status (assigned, unassigned, etc.), contact data for the assignee or delegate, and typically does not require real-time performance as access to this data is not required for ordinary call or session establishment.

Service Data: data necessary to enable service for the TN; it includes addressing data, service features, and so on, and typically does require real-time performance, in so far as this data typically must be queried during call set-up.

Administrative and service data can fit into three categories:

Public: data that anyone can access, for example a list of which numbering resources (unallocated number ranges) are available for acquisition from the Registry.

Semi-restricted: data that a subset of actors can access, for example CSPs may be able to access other CSP’s service data.

Restricted: data that is only available to a small subset of actors, for example a Government Entity may be able access contact information for a User.

While it might appear there are really only two categories, public and restricted based on requestor, the distinction between semi-restricted and restricted is helpful for the use cases below.

2.3. Data Management Architectures

This framework generally assumes that administrative and service data is maintained by CSPs, Registrars, and Registries. The role of a Registry described here is a "thin" one, where the Registry manages basic allocation information for the numbering space, such as information about whether or not the number is assigned, and if assigned, by which Registrar. It is the Registrar that in turn manages detailed administrative data about those assignments, such as contact or billing information for the assignee. In some models, CSPs and Registrars will be composed (the same administrative entity), and in others the Registry and Registrar may similarly be composed. Typically, service data resides largely at the CSP itself, though in some models a "thicker" Registry may itself contain a pointer to the servicing CSP for a number or number block. In addition to traditional centralized Registries, this framework also supports environments where the same data is being managed by multiple administrative entities, and stored in many locations. A distribute registry system is discussed further in [16].

Data store: a service that stores and enables access to administrative and/or service data.

Reference Address: a URL that dereferences to the location of the data store.

Distributed data stores: refers to administrative or service data being stored with multiple actors. For example, CSPs could provision their service data to multiple other CSPs.

Distributed Registries: refers to multiple Registries managing the same numbering resource. Actors could interact with one or multiple Registries. The Registries would update each other when change occurs. The challenge is to ensure there are no clashes, e.g., two Registries assigning the same TN to two different actors.

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 [4] and WEIRDS [12], as well as the TeRI [13] 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 allocations are restricted to any category of service providers, though these tools must and will work in environments with those properties.

The three mechanisms are:

Acquisition: a protocol mechanism for acquiring TNs, including an enrollment process.

Management: a protocol mechanism for associating data with TNs.

Retrieval: a protocol mechanism for retrieving data about TNs.

The acquisition mechanism will enable actors to acquire TNs for use with a communications service. The acquisition mechanism will provide a means to request numbering resources from a service operated by a Registrar, CSP or similar actor. TNs may be requested either on a number-by-number basis, or as inventory blocks. Any actor who grants numbering resources will retain metadata about the assignment, including the responsible organization or individual to whom numbers have been assigned.

The management mechanism will let actors provision data associated with TNs. For example, if a User has been assigned a TN, they may select a CSP to provide a particular service associated with the TN, or a CSP may assign a TN to a User upon service activation. In either case, a mechanism is needed to provision data associated with the TN at that CSP.

The retrieval mechanism will enable actors to learn information about TNs, typically by sending a request to a CSP. For some information,

an actor may need to send a request to a Registry rather than a CSP. Different parties may be authorized to receive different information about TNs.

As an example, a CSP might use the acquisition interface to acquire a chunk of numbers from a Registrar. Users might then provision administrative data associated with those numbers at the CSP through the management interface, and query for service data relating to those numbers through the retrieval interface of the CSP.

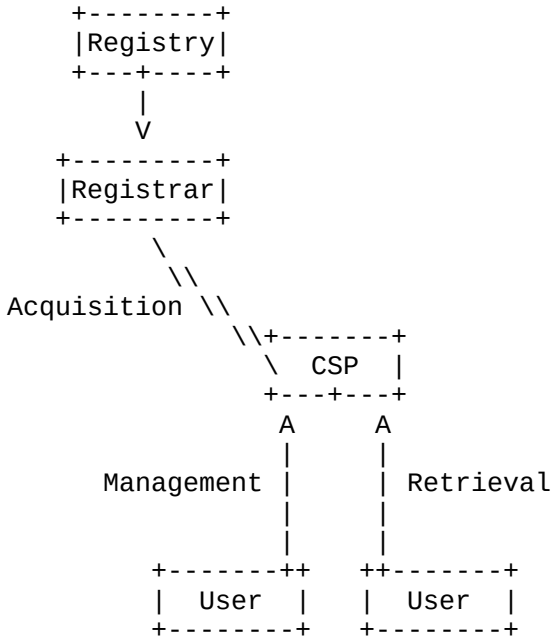


Figure 2: Example of the Three Interfaces

4. Use Cases

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

4.1. Acquisition

There are various scenarios for how TNs can be acquired by the relevant actors: a CSP, Service Enabler, or User. There are three actors from which numbers can be acquired: a Registrar, a CSP and a User (presumably one who is delegating to another party). It is assumed that Registrars are either composed with Registries, or that Registrars have established business relationships with Registries that enable them to distribute the numbers that the Registries here administer. In these use cases, a User may acquire TNs either from a CSP or a Registry, or from an intermediate delegate.

4.1.1. CSP Acquires TNs from Registrar

The most fundamental and traditional numbering use case is one where a CSP, such as a carrier, requests a block of numbers from a Registrar to hold as inventory or assign to customers.

Through some out-of-band business process, a CSP develops a relationship with a Registrar. The Registrar maintains 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 Registrar must authenticate and authorize the CSP, and then either grant or deny a request. When an assignment occurs, the Registry creates and stores administrative information related to the assignment such as TN status and Registrar contact information, and removes the specific TN(s) from the pool of those that are available for assignment. As a part of the acquisition and assignment process, the Registry provides any necessary credentials (for example, STIR certificates [14]) to the Registrar to be used to prove the assignment for future transactions.

Before it is eligible to receive TN 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. TNs assigned to a User are always considered assigned, not inventory. The CSP will associate service information for that TN, e.g., service address, and make it available to other CSPs to enable interoperability. The CSP may need to update the Registrar regarding this service activation (this is part of the "TN status" maintained by the Registrar).

4.1.2. User Acquires TNs from CSP

Today, a User typically acquires a TN from CSP when signing up for communications service or turning on a new device. In this use case, the User becomes the delegate of the CSP.

A User creates or has a relationship with the CSP, and subscribes to a communications service which includes the use of a TN. The CSP collects and stores administrative data about the User. The CSP then activates the User 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 any necessary credentials to the User (for example, a STIR certificate [14]) to prove the assignment for future transactions. Such credential could be delegated from the one provided by the Registrar to the CSP to continue the chain of assignment.

The CSP could assign a TN from its existing inventory or it could acquire a new TN from the Registrar 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 Registrar about the assignment so the Registrar has current assignment data.

4.1.3. CSP Delegates TNs to Another CSP

A reseller or a service bureau might acquire a block of numbers from a CSP to be issued to Users.

In this case, the delegate CSP has a business relationship with the assignee CSP. The assignee CSP collects and stores administrative data about the delegate. The assignee then activates the 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 any necessary credentials to the delegate CSP (for example, a STIR certificate [14]) to prove the assignment for future transactions. Such credentials could be delegated from the one provided by the Registry to the CSP to continue the chain of assignment.

The CSP could assign a block from its existing inventory or it could acquire new TNs from the Registrar 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 Registrar about the assignment so the Registrar has current assignment data. The Delegate may need to provide

utilization and assignment data to the Registry, either directly or through the CSP.

4.1.4. User Acquires TNs from a Delegate

Acquiring a TN from a delegate 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 re-delegating the TNs would be performing functions done by the CSP, e.g., providing any credentials, collecting administrative data, creative service data, and so on.

4.1.5. User Acquires Numbers from Registrar

Today, a user wishing to acquire a freephone number may browse the existing inventory through one or more Registrars, comparing their prices and services. Each such Registrar either is a CSP, or has a business relationship with a CSP to provide services for that freephone number.

Acquiring a TN from a Registrar follows the process in Section 4.1.1, as it should be similar to how a CSP acquires TNs from a Registrar. In this case, the User must establish some business relationship directly to a Registrar, similarly to how such functions are conducted today when Users purchase domain names. For the purpose of status information kept by the Registry, TNs assigned to a User are always considered assigned, not inventory.

In this use case, after receiving a number assignment from the Registrar, 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. Management

The management protocol mechanism is needed to associate administrative and service data with TNs, and may be used to refresh or rollover associated credentials.

4.2.1. Management of Administrative Data

Administrative data is primarily related to the status of the TN, its administrative contacts, and the actors involved in providing service to the TN. Protocol interactions for administrative data will therefore predominantly occur between CSPs and Users to the Registrar, or between Users and delegate CSPs to the CSP.

Most administrative data is not a good candidate for a distributed data store model. Access to it does not require real-time performance therefore local caches are not necessary. And it will include sensitive information such as user and contact data.

Some of the data could lend itself to being publicly available, such as CSP and TN assignment status. In that case it would be deemed public information for the purposes of the retrieval interface.

4.2.1.1. CSP to Registrar

After a CSP acquires a TN or block of TNs from the Registrar (per Section 4.1.1 above), it then provides administrative data to the Registrar as a step in the acquisition process. The Registrar will authenticate the CSP and determine if the CSP is authorized to provision the administrative data for the TNs in question. The Registry will update the status of the TN, i.e., that it is unavailable for assignment. The Registrar will also maintain administrative data provided by the CSP.

Changes to this administrative data will not be frequent. Examples of changes would be terminating service (see Section 4.2.3.2) and changing a CSP or delegate. Changes should be authenticated by a credential to prove administrative responsibility for the TN.

In a distributed Registry model, TN status, e.g., allocated, assigned, available, unavailable, would need to be provided to other Registries in real-time. Other administrative data could be sent to all Registries or other Registries could get a reference address to the host Registry’s data store.

4.2.1.2. User to CSP

After a User acquires a TN or block of TNs from a CSP, the User will provide administrative data to the CSP. The CSP commonly acts as a Registrar in this case, maintaining the administrative data and only notify the Registry of the change in TN status. In this case, the Registry maintains a reference address to the CSP/Registrar’s administrative data store so relevant actors have the ability to access the data. Alternatively a CSP could send the administrative data to an external Registrar to store. If there is a delegate between the CSP and user, they will have to ensure there is a mechanism for the delegate to update the CSP as change occurs.

4.2.1.3. User to Registrar

If the User has a direct relationship with the Registrar, then naturally the user could provision administrative data associated with their TN directly to the Registrar. This is the case, for example, with the freephone example, where a User has a business relationship with its freephone provider, and the freephone provider maintains account and billing data. While delegates necessarily are not assignees, some environments as an optimization might want to support a model where the delegate updates the Registrar directly on changes, as opposed to sending that data to the CSP or through the CSP to the Registrar. As stated already, the protocol should enable Users to acquire TNs directly from a Registrar, which Registrar may or may not also act as a CSP. In these cases the updates would be similar to that described in Section 4.2.1.1.

4.2.2. Management of Service Data

Service data is data required by an originating or intermediate CSP to enable communications service to a User: a SIP URI is an example of one service data element commonly used to route communications. CSPs typically create and manage service data, however it is possible that delegates and Users could as well. For most use cases involving individual Users, it is anticipated that lower-level service information changes would be communicated to CSPs via existing protocols (like the baseline SIP REGISTER [2] method) rather than through any new interfaces defined by MODERN.

4.2.2.1. CSP to other CSPs

After a User enrolls for service with a CSP, in the case where the CSP was assigned the TN by a Registrar, the CSP will then create a service address (such as a SIP URI) and associate it with the TN. The CSP needs to update this data to enable service interoperability. There are multiple ways that this update can occur, though most commonly service data is exposed through the retrieval interface (see Section 4.3. For certain deployment architectures, like a distributed data store model, CSPs may need to provide data directly to other CSPs.

If the CSP is assigning a TN from its own inventory it may not need to perform service data updates as change occurs because the existing service data associated with inventory may be sufficient once the TN is put in service. They would however likely update the Registry on the change in status.

4.2.2.2. User to CSP

Users could also associate service data to their TNs at the CSP. An example is a User acquires a TN from the Registrar (as described in Section 4.1.5) and wants to provide that TN to the CSP so the CSP can enable service. In this case, once the user provides the number to the CSP, the CSP would update the Registry or other actors as outlined in Section 4.2.2.1.

4.2.3. Managing Change

This section will address some special use cases that were not covered in other sections of 4.2.

4.2.3.1. 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. The User provides their credential to the new CSP and the CSP initiates the change in service.

In the simplest scenario, where there’s an authoritative composed Registry/Registrar that maintains service data, the new CSP provides the new service data with the User’s credential to the Registry/ Registrar, which then makes the change. The old credential is revoked and a new one is provided. The new CSP or the Registrar would send a notification to the old CSP, so they can disable service. 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 service data maintained by the CSP must be removed, and similarly, the CSP must delete any such information it provisioned in the Registry.

In a similar model to common practice in some environments today, the User could provide their credential to the old CSP, and the old CSP initiates the change in service.

If there was a distributed Registry that maintained service data, the Registry would also have to update the other Registries of the change.

4.2.3.2. Terminating a Service

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 service data maintained by the

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

The TN will change state from assigned to unassigned, the CSP will update the Registry. Depending on policies the TN could go back into the Registry, CSP, or delegate's pool of available TNs and would likely enter an ageing process.

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

4.3. Retrieval

Retrieval of administrative or service data will be subject to access restrictions based on the category of the specific data; public, semi-restricted or restricted. Both administrative and service data can have data elements that fall into each of these categories. It is expected that the majority of administrative and service data will fall into the semi-restricted category: access to this information may require some form of authorization, though service data crucial to reachability will need to be accessible. In some environments, it's possible that none of the service data will be considered public.

The retrieval protocol mechanism for semi-restricted and restricted data needs a way for the receiver of the request to identify the originator of the request and what is being requested. The receiver of the request will process that request based on this information.

4.3.1. Retrieval of Public Data

Under most circumstances, a CSP wants its communications service to be publicly reachable through TNs, so the retrieval interface supports public interfaces that permit clients to query for service data about a TN. Some service data may however require that the client by authorized to receive it, per the use case in Section 4.3.3 below.

Public data can simply be posted on websites or made available through a publicly available API. Public data hosted by a CSP may have a reference address at the Registry.

4.3.2. Retrieval of Semi-restricted Administrative Data

A CSP is having service problems completing calls to a specific TN, so it wants to contact the CSP serving that TN. The Registry authorizes the originating CSP to access this information. It initiates a query to the Registry, the Registry verifies the requestor and the requested data and Registry responds with the serving CSP and contact data.

Alternatively that information could be part of a distributed data store and not stored at the Registry. In that case, the CSP has the data in a local distributed data store and it initiates the query to the local data store. The local data store responds with the CSP and contact data. No verification is necessary because it was done when the CSP was authorized to receive the data store.

4.3.3. Retrieval of Semi-restricted Service Data

A User on a CSP’s network calls a TN. The CSP initiates a query for service data associated with the TN to complete the call, and will receive special service data because the CSP operates in a closed environment where different CSPs receive different responses, and only authorized CSPs may access service data. The query and response must have real-time performance. There are multiple scenarios for the query and response.

In a distributed data store model each CSP distributes its updated service data to all other CSPs. The originating CSP has the service data in its local data store and queries it. The local data store responds with the service data. The service data can be a reference address to a data store maintained by the serving CSP or it can be the service address itself. In the case where it’s a reference address the query would go to the serving CSP and they would verify the requestor and the requested data and respond. In the case where it’s the service address it would process the call using that.

In some environments, aspects of the service data may reside at the Registry itself (for example, the assigned CSP for a TN), and thus a the query may be sent to the Registry. The Registry verifies the requestor and the requested data and responds with the service data, such as a SIP URI containing the domain of the assigned CSP.

4.3.4. Retrieval of Restricted Data

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 Registrar.

For a case where the CSP delegates a number to the User, the CSP might provision the Registrar (or itself, if the CSP is composed with a Registrar) 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 Registrar or CSP to acquire the necessary data. The interfaces necessary for this will be the same as those described in Section 4.3; the Government Entity will be authenticated, and an authorization decision will be made by the Registrar or CSP under the policy dictates established by the National Authority.

5. Acknowledgments

We would like to thank Henning Schulzrinne for his contributions to this problem statement and framework, and to thank Pierce Gorman for detailed comments.

6. IANA Considerations

This memo includes no instructions for the IANA.

7. Security Considerations

The acquisition, management, and retrieval of administrative and service data associated with telephone numbers raises a number of security issues.

Any mechanism that allows an individual or organization to acquire telephone numbers will require a means of mutual authentication, of integrity protection, and of confidentiality. A Registry as defined in this document will surely want to authenticate the source of an acquisition request as a first step in the authorization process to determine whether or not the resource will be granted. Integrity of both the request and response is essential to ensuring that tampering does not allow attackers to block acquisitions, or worse, to commandeer resources. Confidentiality is essential to preventing eavesdroppers from learning about allocations, including the personally identifying information associated with the administrative or technical contracts for allocations.

A management interface for telephone numbers has similar requirements. Without proper authentication and authorization

mechanisms in place, an attack could use the management interface to disrupt service data or administrative data, which could deny service to users, enable new impersonation attacks, prevent billing systems from operating properly, and cause similar system failures.

Finally, a retrieval interfaces has its own needs for mutual authentication, integrity protection, and for confidentiality. Any CSP sending a request to retrieve service data associated with a number will want to know that it is reaching the proper authority, that the response from that authority has not been tampered with in transit, and in most cases the CSP will not want to reveal to eavesdroppers the number it is requesting or the response that it has received. Similarly, any service answering such a query will want to have a means of authenticating the source of the query, and of protecting the integrity and confidentiality of its responses.

8. Informative References

[1] 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>.

[2] 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>.

[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, DOI 10.17487/RFC6116, March 2011, <http://www.rfc-editor.org/info/rfc6116>.

[4] 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>.

[5] Watson, M., "Short Term Requirements for Network Asserted Identity", RFC 3324, DOI 10.17487/RFC3324, November 2002, <http://www.rfc-editor.org/info/rfc3324>.

[6] 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>.

[7] 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>.

[8] 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>.

[9] Schulzrinne, H., "The tel URI for Telephone Numbers", RFC 3966, DOI 10.17487/RFC3966, December 2004, <http://www.rfc-editor.org/info/rfc3966>.

[10] 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>.

[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, DOI 10.17487/RFC5727, March 2010, <http://www.rfc-editor.org/info/rfc5727>.

[12] Newton, A. and S. Hollenbeck, "Registration Data Access Protocol (RDAP) Query Format", RFC 7482, DOI 10.17487/RFC7482, March 2015, <http://www.rfc-editor.org/info/rfc7482>.

[13] Peterson, J., "A Framework and Information Model for Telephone-Related Information (TeRI)", draft-peterson-modern-teri-00 (work in progress), October 2015.

[14] Peterson, J. and S. Turner, "Secure Telephone Identity Credentials: Certificates", draft-ietf-stir-certificates-06 (work in progress), July 2016.

[15] 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.

[16] Bellur, H. and C. Wendt, "Distributed Registry Protocol", draft-wendt-modern-drip-00 (work in progress), October 2015.

[17] Rosenberg, J. and H. Schulzrinne, "Session Initiation Protocol (SIP): Locating SIP Servers", RFC 3263, DOI 10.17487/RFC3263, June 2002, <http://www.rfc-editor.org/info/rfc3263>.

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: January 9, 2017

J. Peterson
Neustar, Inc.

July 8, 2016

An Architecture and Information Model for Telephone-Related Information (TeRI)
draft-peterson-modern-teri-01

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 related to 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 January 9, 2017.

Copyright Notice

Table of Contents

1. Terminology 3
2. Motivation 3
3. Overview of Operations 5
3.1. Relationship to the MODERN Framework 6
4. The Information Model 7
4.1. Record Elements 8
4.1.1. Identifier 8
4.1.2. Authority 8
4.1.3. Contact 8
4.1.4. Subject 8
4.1.5. Service 8
4.1.6. Signature 9
4.2. Element Value Types 9
4.2.1. Service Types 9
4.2.2. Public Key Type 10
4.2.3. Contact Type 11
4.2.4. Expiry Type 11
4.2.5. Priority Type 11
4.2.6. Record Identifier Type 11
4.2.7. Signature 11
4.2.8. Extension Type 11
5. Operations 11
5.1. Common to All Operations 12
5.1.1. Requests 12
5.1.2. Responses 13
5.2. The Acquisition Operation 14
5.3. The Management Operation 14
5.4. The Retrieval Operation 15
5.5. Common Attributes 15
5.5.1. Administrative Attributes 15
5.5.2. Service Attributes 15

6. Implementing Operations 16
6.1. Transport Independence 16
6.2. Bindings 17
6.3. Encodings 18
6.4. Profiles 19
7. Security Considerations 19
8. IANA Considerations 19
9. Acknowledgements 20
10. Informative References 20
Author's Address 22

1. Terminology

In this document, the key words "MAY", "MUST, "MUST NOT", "SHOULD", and "SHOULD NOT", are to be interpreted as described in [RFC2119]. This document also incorporates the terminology of the MODERN Framework [I-D.ietf-modern-problem-framework].

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]) 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. SIP, for example, quickly developed a convention for using a TEL URI in the user part of its URIs.

The ENUM ([RFC6116]) effort originally 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, differentiating 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 users to rely on telephone 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. Ultimately, the resources identified by telephone numbers must also be reachable on the Internet, and different applications might want to use different protocols to retrieve information about numbers. 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.ietf-modern-problem-framework].

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. Overview of Operations

In TeRI, Clients use Operations to acquire, manage, or retrieve Records, which are typically stored at Services. Every Operation consists of a Request and a Response. Requests may pass directly from a Client to a Service, or they may pass through one or more Request Intermediaries; Request Intermediaries can modify Requests and Responses in transit. A Response will contain a Response Code indicating the status of the requested Operation. Both Requests and Responses can, in certain Operations, carry Records. TeRI does not specify any specific data format or underlying protocol to instantiate Requests, Responses, or Records: TeRI is an abstract architecture that must be implemented with concrete bindings and encodings (see Section 6).

The TeRI information model (see Section 4) specifies the baseline contents of Records, though Records are designed to be extended by future specifications for particular use cases or environments. Records provide information related to telephone numbers; a Record may apply to one telephone number, a block of numbers, or several discrete blocks of numbers. There may be multiple Records stored at a Service which cover a single telephone number: this may include multiple Records that apply only to that one telephone number, which probably have been provisioned by different Authorities, as well as Records applying to a telephone number range which contains that one telephone number. Authorities sign Records, and Clients typically have a trust relationship with those Authorities.

The three TeRI Operations are as follows:

The Acquisition Operation enables a Client to request the allocation of unallocated telephone numbers that are held by a Service on behalf of an Authority. A Service makes an authorization decision before allocating the telephone number(s) in accordance with the policy of the Authority. One or more new Records may be created as a result of a successful Acquisition Operation, and the Service will pass any such Record(s) to the acquiring Client as well as retaining them locally at the Service. As a result of a successful Acquisition Operation, the administrative entity operating the Client will typically become a new Authority for the allocated telephone numbers.

The Management Operation enables a Client to push new values for a Record to a Service. In the baseline Operation described in this document, the Client pushes the entire value of the Record to the Service. The Service then makes an authorization decision to determine whether or not the Client is permitted to upload the Record in question. The policy behind those authorization

decisions is outside the scope of this document, though at a high-level, the Client must be an Authority for a telephone number in order to publish and modify Records associated with that number. However, outside of hierarchical Authorities, Clients will not be able to modify or delete Records related to that number that have been provisioned by other Authorities.

The Retrieval Operation enables a Client to request one or more Records that are stored at a Service. Some Records may contain public information, and some may contain information that requires an authorization decision to be made before it is shared with a Client. Note that Services may have trust relationships with Request Intermediaries, and that the Response may depend on that trust relationship rather than on the Service's trust relationship with the Client. Although a Client acquires Records from a Service, a client need not have a trust relationship with it - typically, the Client trusts the Record because it trusts the Authority which signed the Record.

All entities that act as TeRI Services will offer at least the Management and Retrieval interfaces, and some will also offer the Acquisition interface. All entities that act as TeRI Clients will implement at least the Retrieval Operation; others may implement the client side of one or both of the Management and Acquisition Interfaces.

3.1. Relationship to the MODERN Framework

The MODERN Framework [I-D.ietf-modern-problem-framework] enumerates a series of actors and use cases related to telephone number administration on the Internet. In terms of actors, it details interactions between Users, Communications Service Providers (CSPs), Registries, Registrars, and Government Entities. These actors implement the interfaces and Operations of TeRI Clients or Services in support of various use cases. Typically, Users, CSPs, and Government Entities act as TeRI Clients, and CSPs, Registries, and Registrars act as TeRI Services.

In the MODERN framework, the lifecycle of a number begins with a Registry. Registrars acquire telephone numbers from Registries, and make those numbers available for allocation. Thus, an Acquisition Operation is used by a Registrar that acquires numbers from a Registry, and this Request, if successful, will result in the creation of a Record that is returned in the Response. That Record renders the Registrar an Authority for the telephone numbers in question, but that Record will contain exclusively Administrative Data, with no Service Data.

In some cases, that Registrar will also fulfil the role of a CSP, and as a CSP, it will allocate those numbers to Users and generate any associated Records itself. Alternatively, a Registrar that does not act as a CSP may in turn act as a TeRI Service to which CSPs, and potentially Users, will send Acquisition Requests to acquire number blocks or individual numbers. Through that process, CSPs and Users can also become Authorities for telephone numbers. New Records containing Administrative Data indicating the contact information and so forth of the CSP or the User will be generated when that allocation occurs; those Records will be stored at the Registrar. The Registrar may also house a "glue" Record of Service Data that indicates the servicing CSP for the telephone number, and in particular the Retrieval interface of that CSP where Records with further Service Data can be found.

The Authorities who create and propagate Records of Service Data are typically CSPs and Users. Most commonly, CSPs will store these Service Data Records, and make them accessible through a Retrieval interface. CSPs may also propagate these Records to various external directories; the signature of the CSP and expiry data in the Record will prove its integrity and freshness to any relying party. It is envisioned that multiple Authorities may create Records for different services that are associated with a given telephone number.

Finally, CSPs and Users may query a Retrieval interface at a CSP to acquire Records containing Service Data that will enable them to route communications. The Retrieval interface will enable Clients to ask for Records associated with particular services, though Retrieval can present Clients with a number of service options. Entities may also query the Retrieval Interface of Registrars to acquire Administrative Data about a telephone number, though it is likely that authorization policies will restrict access to that data. Government Entities may have legal relationships with Registrars that grant them authorization privileges with regard to Administrative Data.

4. 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 that encompasses a particular telephone number. Under various circumstances detailed in Section 5, 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.

4.1. Record Elements

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

4.1.1. 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.ietf-modern-problem-framework]).

4.1.2. 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.

4.1.3. 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.ietf-modern-problem-framework]).

4.1.4. Subject

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

4.1.5. Service

Records optionally have one or more Service entries. A Service may be of any Service Type, as given in Section 4.2.1.

4.1.5.1. 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.

4.1.5.2. 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.

4.1.6. 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]

4.2. 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.

4.2.1. Service Types

4.2.1.1. 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.]

4.2.1.1.1. 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

4.2.1.2. 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

4.2.1.3. 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

4.2.1.4. 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

4.2.1.5. Trunk Group Type

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

Type Code: G

4.2.1.6. Service Provider Identifier (SPID) Type

The SPID type consists of a four-digit number.

[TBD - introduce other elements for alternative SPID syntaxes]

Type Code: ?

4.2.2. Public Key Type

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

Type Code: C

4.2.3. Contact Type

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

Type Code: C

4.2.4. Expiry Type

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

Type Code: E

4.2.5. 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

4.2.6. Record Identifier Type

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

Type Code: U

4.2.7. Signature

[Syntax TBD]

Type Code: S

4.2.8. Extension Type

This code is reserved for future use.

Type Code: X

5. Operations

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

5.1. 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 of failure.

5.1.1. 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 contain the Identifier of the Record they concern, and may convey that in an Attribute; others will query for all Records matching a given Subject.

5.1.1.1. 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.

5.1.1.1.1. 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.

5.1.1.1.2. 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.

5.1.1.2. 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.

5.1.1.2.1. Attributes

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

5.1.2. 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.

5.1.2.1. 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]

5.2. 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.

5.3. 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 Response 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. In almost all circumstances, however, the Service will locate that Record(s) that a

Management Request modifies through the Identifier attribute of each Record in the Management Request.

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 authorize whether or not the Client is authorized to modify the Record in question via the Source of the Management Request.

5.4. 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, or the Attributes otherwise qualify the Request. Typically an Attribute will specify a Service or Service Type that the Client seeks Records for.

Successful Retrieval Responses always contain one or more Records; unsuccessful Responses never contain Records.

5.5. 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 4.2.

5.5.1. Administrative Attributes

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

5.5.2. Service Attributes

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

5.5.2.1. 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.

6. 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.

6.1. 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.

6.2. 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.

6.3. 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]

6.4. 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.

7. 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 6.1.

[TBD - more]

8. 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]

9. Acknowledgements

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

10. Informative References

[I-D.ietf-modern-problem-framework] Peterson, J. and T. McGarry, "Modern Problem Statement, Use Cases, and Framework", draft-ietf-modern-problem-framework-00 (work in progress), April 2016.

[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: January 9, 2017

H. Bellur
C. Wendt, Ed.
Comcast

July 8, 2016

Distributed Registry Protocol
draft-wendt-modern-drip-01

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 January 9, 2017.

Copyright Notice

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
4.7.1. API - POST /heartbeat/node/:nodeid 14
4.8. Key-Value Data Update Entitlement Verification 15
5. Security Considerations 15
5.1. HTTPS 15
5.2. Authorization 15
5.3. Payload Validation 15
6. References 15
Authors' Addresses 16

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.

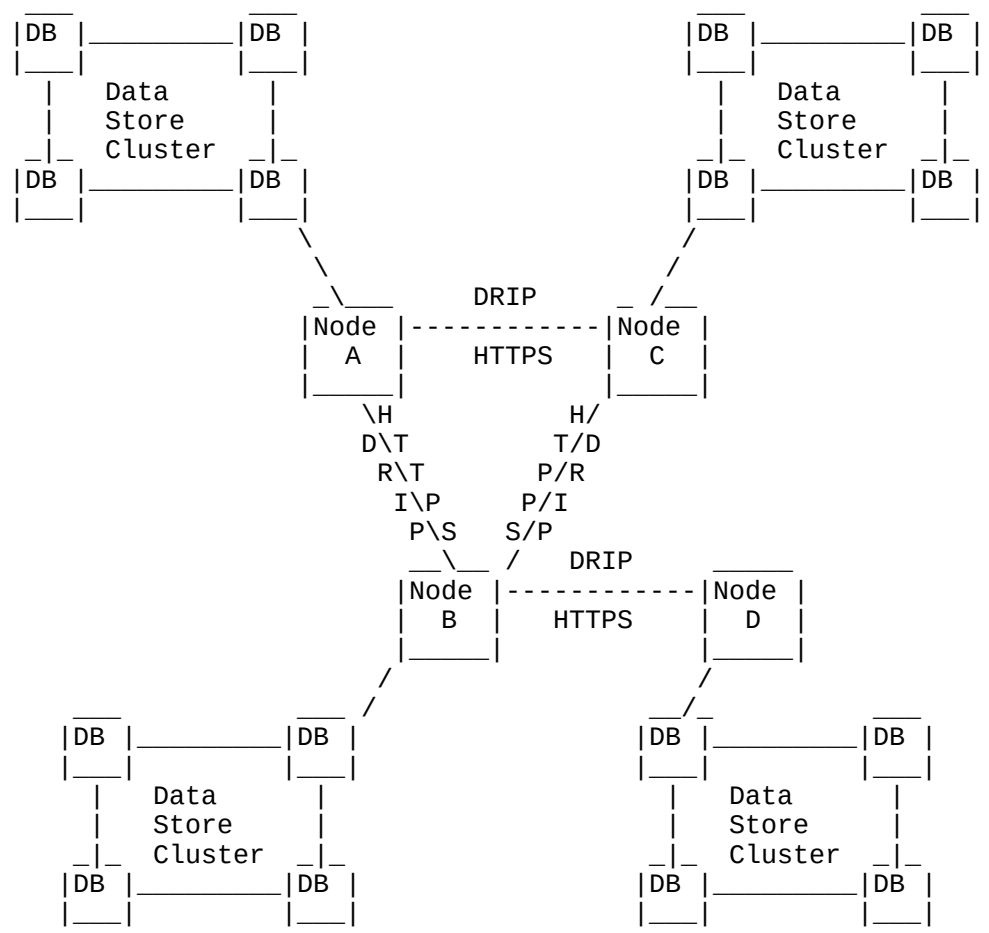
Receiver Node:
A node that forwards the propagated key-value data.

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 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 it’s 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

Example (using cURL)

Request

```
$ curl -i -H "DRiP-Node-ID: nodeA" -H "Authorization: eyJ0e..."
    -X POST https://nodearegistry.com/node/nodeA/active
```

Response

```
HTTP/1.1 200 OK
```

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

Example (using cURL)

Request

```
$ curl -i -H "DRiP-Node-ID: nodeA" -H "Authorization: eyJ0e..."
    -X POST https://nodearegistry.com/node/nodeA/inactive
```

Response

```
HTTP/1.1 200 OK
```

4.2.3. API - 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 (using cURL)

Request

```
$ curl -i -H "DRiP-Node-ID: nodeA" -H "Authorization: eyJ0e..."
    -X GET https://nodearegistry.com/state
```

Response

HTTP/1.1 200 OK with the following JSON object.

Property	Description
state	"active" or "inactive" or "sync"

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 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:

- 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.

- 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.

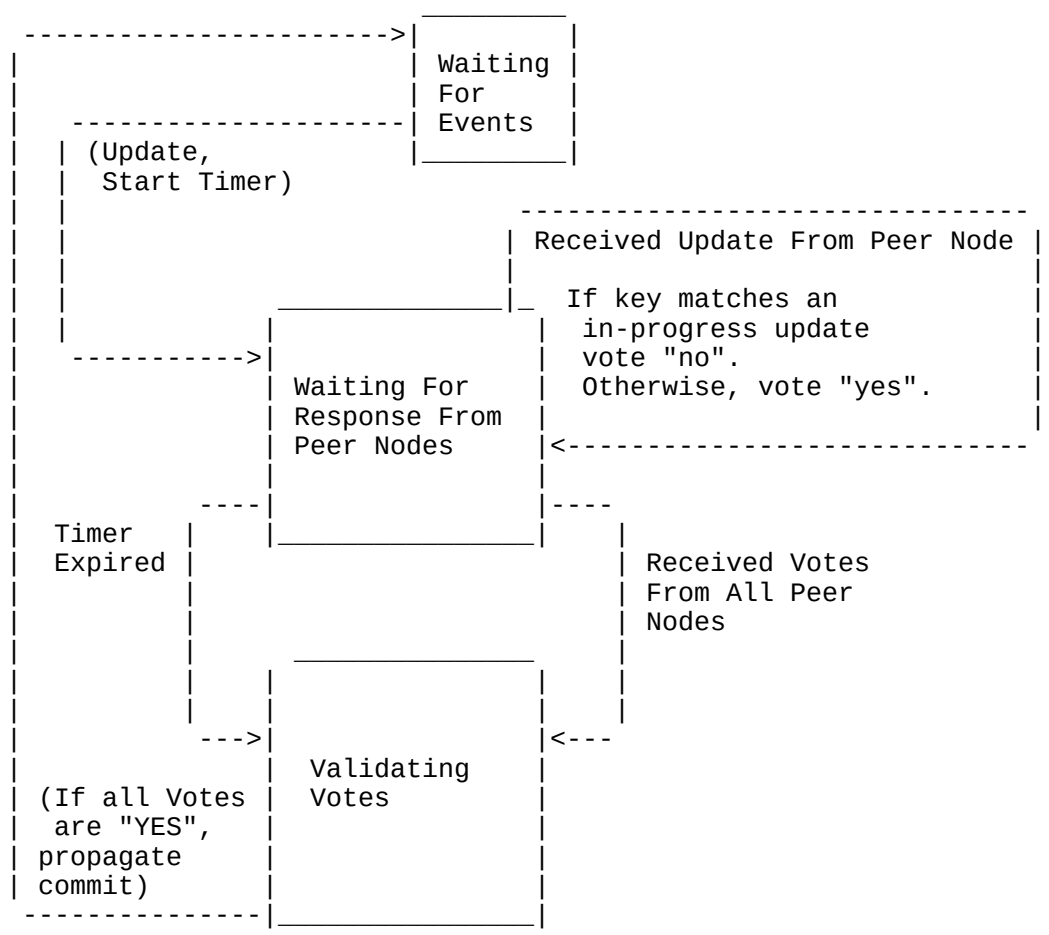
- If DRiP-Node-ID field value matches with a stored node ID and DRiP-Node-Counter-reset field value is true:

  * 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.

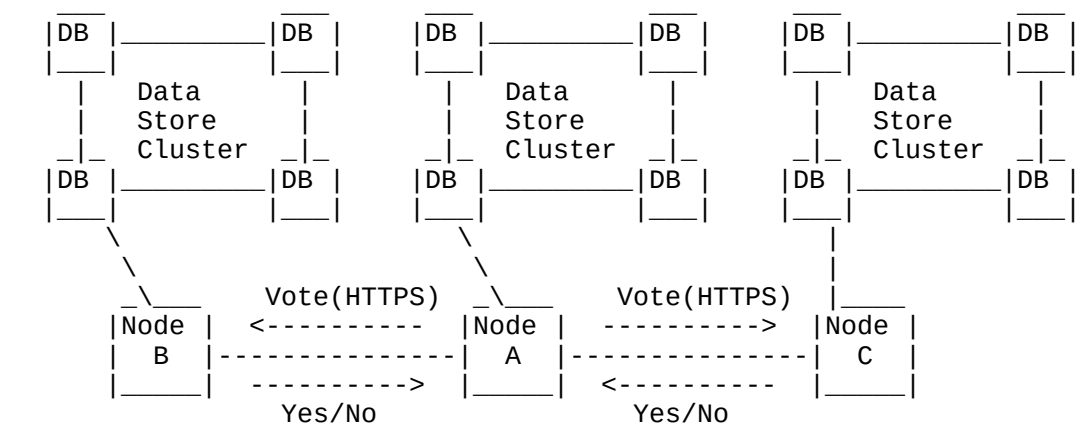
The Initiator node MUST set a timeout period to get response from its peer nodes.

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.



Voting Phase

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

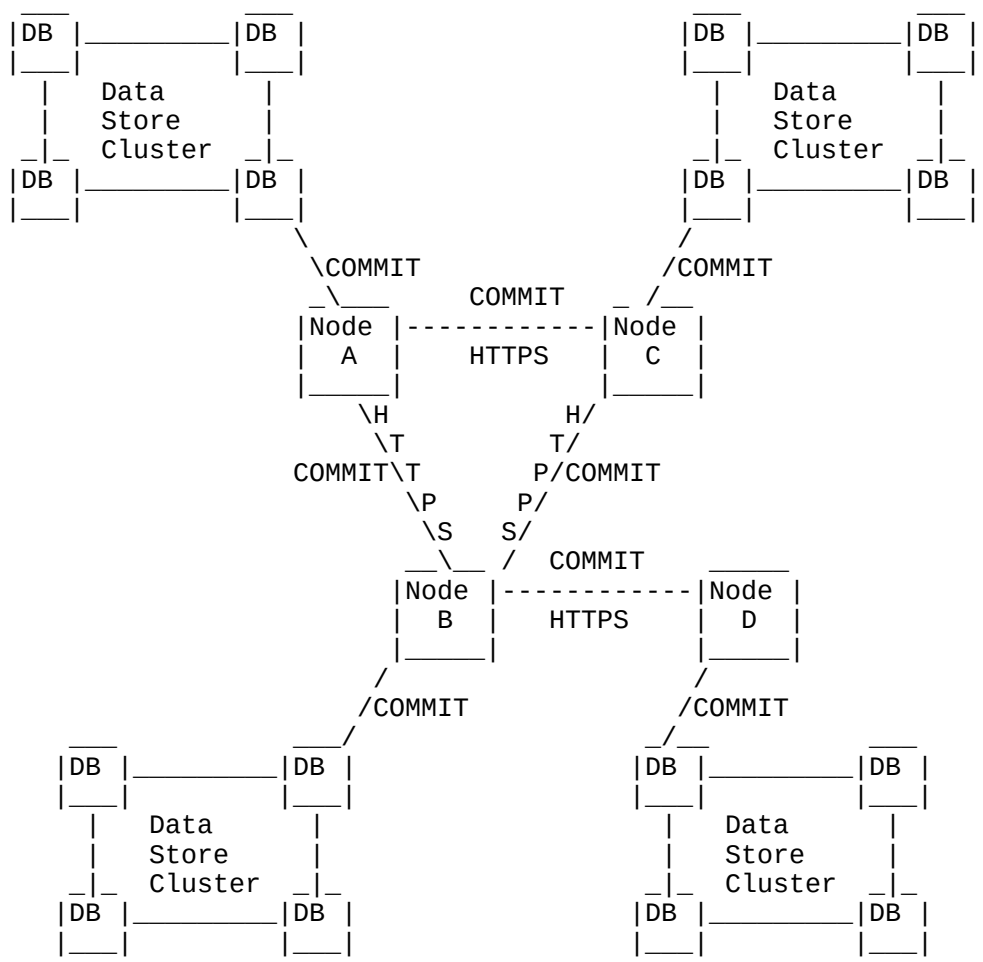
```
$ curl -i -X POST http://nodearegistry.com/node/nodeA/response/yes
```

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

```
$ curl -i  -H "DRiP-Node-ID: nodeA" -H "Authorization: eyJ0e..."
        -X POST https://peernode.com/sync/node/nodeA
```

Response

HTTP/1.1 200 OK

4.7. Heartbeat

Periodic heartbeats are required for a node to determine it’s visibility to the rest of it’s peer nodes and whether it should put itself in "inactive" mode. The proceedure for heartbeats is as follows.

A node sends periodic heartbeat requests to its peer nodes with an indication of its state. These heartbeat requests are not to be propagated beyond the peer nodes.

If all of its peer nodes cannot be reached or do not respond with 200 OK, then the node that sent the heartbeat request will set its own state to "inactive". This is based on the reasonable assumption that none of the peer nodes are able to communicate with this node until a new heartbeat request is sucessful. Once in the inactive state, the node will

- not propagate any incoming key-value data
- not update any incoming key-value data
- continue to send the periodic heartbeat requests to its peer nodes. If any one responds with 200 OK, then the node will move its state to "synchronizing" and will re-synchronize its data with any active peer node as detailed in section 4.6

In addition, any one or more peer nodes that cannot be reached or did not respond with 200 OK should not be used to propagate key-value data until it responds (with 200 OK) to the heartbeat request.

4.7.1. API - POST /heartbeat/node/:nodeid

Example (using cURL)

Request

```
$ curl -i  -H "DRiP-Node-ID: nodeA" -H "Authorization: eyJ0e..."
        -X POST -d ’<state>’ https://peernode.com/heartbeat/node/nodeA
```

Response

HTTP/1.1 200 OK

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. Authorization

All nodes MUST validate their authority to consume the HTTP APIs of a peer node by adding a JSON Web Token (JWT) value [RFC7519] in the Authorization request-header field.

The creation and verification of the JWT should be based on a digital signature. For most distributed registry scenerios where the owner of a node may not have a direct relationship with another node owner, a PKI based certificate approach is highly suggested. For protection against replay attacks, the claim set SHOULD contain an "iat" claim and the signature should be verified to be signed by the expected owner of the peer node. The "iat" claim identifies the time at which the JWT was issued and can be used to validate when the time of the transaction occurred.

5.3. Payload Validation

In addition to the DRiP level protocol protection, it is highly suggested to sign and validate part or all of the JSON update payloads to the originator of the update. DRiP does not define anything regarding the contents of the payload, so this document does not address this in any way.

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>.

[RFC7519] Jones, M., Bradley, J., and N. Sakimura, "JSON Web Token (JWT)", RFC 7519, DOI 10.17487/RFC7519, May 2015, <http://www.rfc-editor.org/info/rfc7519>.

Authors' Addresses

Harsha Bellur
Comcast
One Comcast Center
Philadelphia, PA 19103
USA

Email: Harsha_Bellur@cable.comcast.com

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

Email: chris-ietf@chriswendt.net