dispatch	J. Rosenberg
Internet-Draft	jdrosen.net
Intended status: Standards Track	C. Jennings
Expires: September 9, 2010	Cisco
	March 08, 2010

TOC

Verification Involving PSTN Reachability: Requirements and Architecture Overview
draft-rosenberg-dispatch-vipr-overview-02

Abstract

The Session Initiation Protocol (SIP) has seen widespread deployment within individual domains, typically supporting voice and video communications. Though it was designed from the outset to support inter-domain federation over the public Internet, such federation has not materialized. The primary reasons for this are the complexities of inter-domain phone number routing and concerns over security. This document reviews this problem space, outlines requirements, and then describes a new model and technique for inter-domain federation with SIP, called Verification Involving PSTN Reachability (ViPR). ViPR addresses the problems that have prevented inter-domain federation over the Internet. It provides fully distributed inter-domain routing for phone numbers, authorized mappings from phone numbers to domains, a new technique for automated VOIP anti-spam, and privacy of number ownership, all while preserving the trapezoidal model of SIP.

Status of this Memo

This Internet-Draft is submitted to IETF in full conformance with the provisions of BCP 78 and BCP 79.

Copyright Notice

Table of Contents

---

1. Introduction

TOC

The Session Initiation Protocol (SIP) was originally published as RFC 2543 (Handley, M., Schulzrinne, H., Schooler, E., and J. Rosenberg, "SIP: Session Initiation Protocol," March 1999.) [RFC2543] in May of 1999. This was followed by subsequent publication of RFC 3261 (Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., Handley, M., and E. Schooler, "SIP: Session Initiation Protocol," June 2002.) [RFC3261], which brought the protocol to sufficient maturity to enable large scale market adoption.
And indeed, it has seen large scale market adoption. SIP has seen hundreds of implementations, spanning consumer products, enterprise servers, and large scale carrier equipment. It carries billions and billions of minutes of calls, and has become the lingua franca of interconnection between products from different vendors. If one measures success in deployment, then clearly SIP is a success.

However, in other ways, it has failed. SIP was designed from the ground up to enable communications between users in different domains, all over the public Internet. The intention was that real-time communications should be no different than email or the web, with the same any-to-any connectivity that has fueled the successes of those technologies. Though SIP is used between domains, it is typically through private federation agreements. The any-to-any Internet federation model envisioned by SIP has not materialized at scale.
This document introduces a new technology, called Verification Involving PSTN Reachability (ViPR), that enables us to break down the barriers that have prevented inter-domain VoIP. By stepping back and changing some of the most fundamental assumptions about federation, ViPR is able to address the key problems preventing its deployment. ViPR focuses on incremental deployability over the unrealizable nirvana. At the same time, ViPR ensures that SIP's trapezoidal model - direct federation between domains without any intermediate processing beyond IP transport - is realized. That model is required in order to allow innovative new services to be deployed.

---

2. Problem Statement

TOC

The first question that must be asked is this - why haven't we seen widespread adoption of inter-domain SIP federation?
There are many reasons for it. They are - in order of importance - the phone number routing problem, the open pinhole problem, the quality of service problem, and the troubleshooting problem. The two former ones are the most significant.

2.1. The Phone Number Routing Problem

TOC

Inter-domain federation requires that the sending domain determine the address of the receiving domain, in the form of a DNS name (example.com) or one or more IP addresses that can be used to reach the domain. In email and in the web, this is easy. The identifiers used by those services - the email address and web URL respectively - embed the address of the receiving domain. A simple DNS lookup is all that is required to route the connection. SIP was designed to use the same email-style identifiers.
However, most SIP deployments utilize phone numbers, and not email-style SIP URI. This is due to the huge installed base of users that continue to exist solely on the public switched telephone network (PSTN). In order to be reached by users on the PSTN, and in order to reach them, users in SIP deployments need to be assigned a regular PSTN number. Users in SIP deployments need to place that PSTN number on

business cards, use it in their email signatures, and in general, give it out to their friends and colleagues, in order to be reached. While those users could additionally have an email style SIP URI, the PSTN number serves as a single, global identifier that works for receiving calls from users on the PSTN as well as users within the same SIP domain. Why have two identifiers when one will suffice? The universality of PSTN numbers is the reason why most SIP deployments continue to use them - often exclusively.

Another reason is that many SIP deployments utilize hardphones or telephony adaptors, and the user interfaces on these devices - patterned after existing phones - only allow phone-number based dialing. Consequently, these users are only allocated PSTN numbers, and not email-style SIP URI.

Finally, a large number of SIP deployments are in domains where the endpoints are not IP. Rather, they are circuit based devices, connected to a SIP network through a gateway. SIP is used within the core of the network, providing lower cost transit, or providing add-on services. Clearly, in these deployments, only phone numbers are used.

Consequently, to make inter-domain federation incrementally deployable and widely applicable, it needs to work with PSTN numbers rather than email-style SIP URI. Telephone numbers, unlike email addresses, do not provide any indication of the address of the domain which "owns" the phone number. Indeed, the notion of phone number ownership is somewhat cloudy. Numbers can be ported between carriers. They can be assigned to a user or enterprise, and then later re-assigned to someone else. Numbers are granted to users and enterprises through a complex delegation process involving the ITU, governments, and telecommunications carriers, often involving local regulations that vary from country to country.

Therefore, in order to deploy inter-domain federation, domains are required to utilize some kind of mechanism to map phone numbers to the address of the domain to which calls should be routed. Though several techniques have been developed to address this issue, none have achieved large-scale Internet deployments.

2.2. The Open Pinhole Problem

TOC

The inter-domain federation mechanism built into SIP borrows heavily from email. Each domain runs a SIP server on an open port. When one domain wishes to contact another, it looks up the domain name in the DNS, and connects to the that server on the open port. Here, "open" means that the server is reachable from anywhere on the public Internet, and is not blocked by firewalls.

This simple design worked well in the early days of email. However, the email system has now become plagued with spam, to the point of becoming useless. Administrators of SIP domains fear - rightfully so - that if

they make a SIP server available for anyone on the Internet to contact, it will open the floodgates for VoIP spam, which is far more disruptive than email-based spam [RFC5039] (Rosenberg, J. and C. Jennings, “The Session Initiation Protocol (SIP) and Spam,” January 2008.).
Administrators also worry - rightfully so - that an open server will create a back-door for denial-of-service and other attacks that can potentially disrupt their voice service. Administrators are simply not willing to take that risk; rightly or wrongly, voice deployments demand higher uptimes and better levels of reliability than email, especially for enterprises.
Fears around spam and denial-of-service attacks, when put together, form the "open pinhole problem" - that domains are not willing to enable SIP on an open port facing the Internet.
To fix this, a new model for federation is needed - a model where these problems are addressed as part of the fundamental design, and not as an after-thought.

2.3. Quality of Service Problem

TOC

The Internet does not provide any QoS guarantees. All traffic is best effort. This is not an issue for data transaction services, like web and email. It is, however, a concern when using real-time services, such as voice and video.
That said, there are a large number of existing VoIP deployments that run over the Internet. Though the lack of QoS is a concern, it has not proven a barrier to deployment. We believe that, if the more fundamental issues - the phone number routing and open pinhole problems - can be addressed, the QoS problem will sort itself out. As such, we do not discuss this issue further here.

2.4. Troubleshooting Problem

TOC

The final problem that is stopping large scale inter-domain federation is the troubleshooting problem. When connecting calls between domains, problems will happen. Calls will get blocked. Calls will get misdelivered. Features won't work. There will be one-way media or no media at all. The video won't start. Call quality will be poor.
These problems are common in VoIP deployments, and they are tough to troubleshoot even within a single administrative domain. When real-time services extend inter-domain, the problem becomes worse. A new angle is introduced: the first step is identifying who is at fault.
Fortunately, work is underway to improve the ability for network administrators to diagnose VoIP problems. Common log formats [I-D.roach-sipping-clf-syntax] (Roach, A., “Binary Syntax for SIP

Common Log Format," May 2009.) and consistent session IDs [I-D.kaplan-sip-session-id] (Kaplan, H., "A Session Identifier for the Session Initiation Protocol (SIP)," March 2009.), for example, can help troubleshoot interdomain calls.
In addition to these, any new technology that facilitates inter-domain federation needs to have troubleshooting built-in, so that it is not a barrier to deployment.

---

3. Summary of Existing Solutions

TOC

Given the value that inter-domain SIP federation brings, it is no suprise that many attempts have been made at solving it. Indeed, these have all been deployed to varying degrees. However, all of them have fundamental limitations that have inhibited widespread deployment.

3.1. Domain Routing

TOC

The first solution that has been proposed for SIP inter-domain federation is built into SIP itself - domain routing. In this technique, users utilize email-style SIP URI as identifiers. By utilizing the DNS lookup mechanism defined in [RFC3263] (Rosenberg, J. and H. Schulzrinne, "Session Initiation Protocol (SIP): Locating SIP Servers," June 2002.), SIP enables calls to be routed between domains in much the same way email is routed between domains.
This technique works well in theory, but it has two limitations which have limited its deployment:

1. The majority of SIP deployments utilize phone numbers, often exclusively. In such a case, domain routing cannot be used.

2. Domain federation brings with it the possibility (and strong likelihood) of the same levels of spam and DoS attacks that have plauged the email system.

These issues have already been discussed above.

3.2. Public ENUM

TOC

Public ENUM, defined in [RFC3761] (Faltstrom, P. and M. Mealling, "The E.164 to Uniform Resource Identifiers (URI) Dynamic Delegation Discovery System (DDDS) Application (ENUM)," April 2004.), tries to address the phone number routing problem by cleverly placing phone

numbers into the public DNS. Clients can then perform a simple DNS lookup on a phone number, and retrieve a SIP URI which can be used to route to that phone number.

Unfortunately, public ENUM requires that the entries placed into the DNS be populated following a chain of responsibility that mirrors the ownership of the numbers themselves. This means that, in order for a number to be placed into the DNS, authorization to do so must start with the ITU, and from there, move to the country, telecom regulator, and ultimately the end user. The number of layers of beaurocracy required to accomplish this is non-trivial. In addition, the telecom operators - which would be partly responsible for populating the numbers into the DNS - have little incentive to do so. As a consequence, public ENUM is largely empty, and is likely to remain so for the forseeable future.

Instead, ENUM has morphed into a technique for federation amongst closed peering partners, called private ENUM or infrastructure ENUM [RFC5067] (Lind, S. and P. Pfautz, “Infrastructure ENUM Requirements,” November 2007.). While there is value in this technology, it does not enable the open federation that public ENUM was designed to solve.

It is clear from the legacy of ENUM deployments, that any kind of phone number routing solution should not rely on government or telecom processes for population of the databases.

3.3. Private Federations

TOC

Private federations are a cooperative formed amongst a small number of participating domains. The cooperative agrees to use a common technique for federation, and through it, is able to connect to each other. There are many such federations in use today.

Some of these federations rely on a central database, typically run by the federation provider, that can be queried by participating domains. The database contains mappings from phone numbers to domains, and is populated by each of the participating domains, often manually. Each domain implements an agreed-upon query interface that can be used to access the database when a number is called. Sometimes ENUM is used for this interface (called private ENUM), other times, a SIP redirection is used. Some federations also utilize private IP networks in order to address QoS problems. "SIP trunking" - a service being offered by many telecom operators as a SIP-based PRI replacement - is a form of private federation.

Private federations work, but they have one major limitation: scale. As the number of participating domains grows, several problems arise. Firstly, the size of the databases become unruly. Secondly, the correctness of the database becomes an issue, since the odds of misconfigured numbers (either intentionally or accidentally) increases. As the membership grows further, the odds increase that "bad" domains

will be let in, introducing a source of spam and further problems. The owner of the federation can - and often does - assume responsibility for this, and can attempt to identify and shut down misbehaving participants. Indeed, as the size of the federations grow, the owner of the federation needs to spend increasing levels of capital on maintaining it. This, in turn, requires them to charge money for membership, and this can be a barrier to entry.

---

4. Key Requirements

TOC

From the discussion on the problems of inter-domain federation and the solutions that have been attempted so far, several key requirements emerge:

REQ-1: The solution should allow for federation between any number of domains.

REQ-2: The solution must enable users in one domain to identify users in another domain through the use of their existing E.164 based phone numbers.

REQ-3: The solution must work with deployments that utilize any kind of endpoint, including non-IP phones connected through gateways, IP softphones and hardphones.

REQ-4: The solution should not require any change in user behavior. The devices and techniques that users have been using previously to make inter-domain calls should continue to work, but now result in inter-domain IP federation.

REQ-5: The solution should work worldwide, for any domain anywhere.

REQ-6: The solution should not require any new services from any kind of centralized provider. A domain should be able, of its own free-will and accord, to deploy equipment and connect to the federation.

REQ-7: The solution should not require any prior arrangement between domains in order to facilitate federation between those

domains. Federation must occur opportunistically - connections established when they can be.

REQ-8: The solution must work for domains of any size - starting at a single phone to the largest telecom operator with tens of millions of numbers.

REQ-9: The solution must have built-in mechanisms for preventing spam and DoS attacks. This mechanism must be fully automated.

REQ-10: The solution must not require any processing whatsoever by SIP or RTP intermediaries. It must be possible for a direct SIP connection to be established between participating domains.

These requirements, when put together, appear to be mutually unsolvable. And indeed, they have been - until now.

---

5. Executive Overview

TOC

Verification Involving PSTN Reachability (ViPR) is a new technology that is aimed at solving the problems that have prevented large-scale Internet-based SIP federation of voice and video. ViPR solves these problems by creating a hybrid of three technologies - the PSTN itself, a P2P network, and SIP. By combining all three, ViPR enables an incrementally deployable solution to federation.

5.1. Key Properties

TOC

ViPR has several important properties that enable it to solve the federation problem:

Works With Numbers: ViPR enables federation for existing PSTN numbers. It does not require users or administrators to know or configure email-style identifiers. It does not require the allocation of new numbers. It does not require a change in user behaviors. Whatever way users were dialing numbers yesterday, works with ViPR tomorrow.

Works with Existing Endpoints: ViPR does not require any changes to endpoints. Consequently, it works with existing SIP endpoints, or with non-IP endpoints connected through gateways.

Fully Distributed: ViPR does not require any kind of central authority or provider. A domain wishing to utilize ViPR just

deploys it on their own. ViPR utilizes the existing PSTN and existing Internet connectivity the domain already has, and by combining them, achieves inter-domain federation. Domains do not need to wait for their service providers to roll out any kind of new features, databases, or functionality.

Verified Mappings: The biggest issue in mapping from a phone number to a domain or IP address, is determining whether the mapping is correct. Does that domain really own the given phone number? While solutions like ENUM have solved this problem by relying on centralized delegations of authorization, ViPR provides a secure mapping in a fully distributed way. ViPR guarantees that phone calls cannot be misrouted or numbers stolen.

Worldwide: ViPR works worldwide. Any domain that is connected to both the PSTN and the Internet can participate. It doesn't matter whether the domain is in Africa, the Americas, or Australia. Since ViPR does not depend on availability of any regional services beyond IP and PSTN access - both of which are already available globally - ViPR itself is globally available.

Unlimited Scale: ViPR has nearly infinite scale. Any number of domains can participate.

Self-Scale: ViPR self-scales. This means that the amount of computation, memory, and bandwith that a domain must deploy scales in direct proportion to the size of their own user base.

Self-Learning: ViPR is completely automated. A domain never, ever has to configure any information about another domain. It never has to provision IP addresses, domain names, certificates, phone number prefixes or routing rules. Without any prior coordination, ViPR enables one domain to connect to a different domain.

Automated Anti-Spam ViPR comes with a built-in mechanism for preventing VoIP spam. This mechanism is new, and specific to VoIP. In this way, it is fundamentally different from existing VoIP anti-spam technqiues which borrow from email [RFC5039] (Rosenberg, J. and C. Jennings, “The Session Initiation Protocol (SIP) and Spam,” January 2008.). This new technique is fully automated, and requires no configuration by administrators and no participation from end users. Though it is not a 100% solution to the problem, it brings substantial economic and legal ammunition to the table to act as a good deterrent for a long while.

Feature Velocity: ViPR enables direct SIP connections between two domains seeking to federate. There are no SIP intermediaries of any sort between the two. This means that domains have no dependencies on intermediaries for deployment of new features.

Designed for the Modern Internet:

> ViPR is built to run on the modern Internet. It assumes the worst from everyone. It assumes limited connectivity. It assumes network failures. It assumes there are attackers seeking to eavesdrop calls. Security is built-in and cannot be disabled.

Reliable: ViPR is reliable. Through its hybridization of the PSTN and the Internet, it makes sure that calls always go through. Indeed, to route a call between domains A and B, ViPR never depends on a server or service anywhere outside of domains A and B (besides vanilla PSTN and IP access) being operational.

At first glance, these properties seem impossible to realize. And indeed, given the assumptions that have traditionally been made about how federation has to work, these properties are impossible to realize. It is only by stepping back, and rethinking these fundamental assumptions, that a solution can be found.

5.2. Challenging Past Assumptions

TOC

Two unstated assumptions of SIP federation are challenged by ViPR. The first assumption that federation solutions have made is this:

> The purpose of SIP federation is to eliminate the PSTN, and consequently, we cannot assume the PSTN itself as part of the solution.

Though unstated, this assumption has clearly been part of the design of existing solutions. SIP federation based on email-style URIs, as defined in RFC 3261, doesn't utilize or make mention of the PSTN. Solutions like ENUM, or private registries, do not utilize or make mention of the PSTN. In one sense, it's obvious that they shouldn't - after all, the purpose is to replace the PSTN. However, such an approach ignores an incremental solution - a solution which utilizes the PSTN itself to solve the hard problems in SIP federation. After all, the PSTN has accomplished a great deal. It reaches worldwide. It provides a global numbering translation service that maps phone numbers to circuits. It is highly reliable, and provides QoS. It has been built up over decades to achieve these goals. This begs the question - can we build upon the capabilities already provided by the PSTN, and use them to solve the problems that plague SIP federation? Indeed, the answer is yes once another assumption is challenged. This second assumption is:

> A federation solution must be the same as the final target federation architecture, and not just a step towards it.

Though unstated, this assumption has also been true. SIP's email-style federation was a pure 'target architecture' - the place we want to get to. ENUM was the same - a worldwide global DNS database with everyone's phone numbers - an unrealizable nirvana of open connectivity. Historically, technologies are more successful when they are incrementally deployable. Indeed, in many cases, the target architecture is unrealizable because there is no obvious way to get there. As such, the focus needs to be on the next incremental step that we can take, and that step in turn creates the technological and market pressures that will drive the next step. In the end, the target may not be the perfect nirvana we all imagined, but we've at least arrived. As such, ViPR is very much focused on incremental deployability. It is not the end of the federation story, it is the beginning. It discards the nirvana of perfect IP federation for a solution that federates most, but not all calls, by relying on the PSTN to fill in the gaps. ViPR's philosophy is not to let the perfect be the enemy of the good.

5.3. Technical Overview

TOC

A high level view of the architecture is shown in Figure 1 (High Level Architecture). The figure shows four different domains, a.com, b.com, c.com and d.com, federating using ViPR technology. Each domain is connected to both the public Internet and to the traditional PSTN.

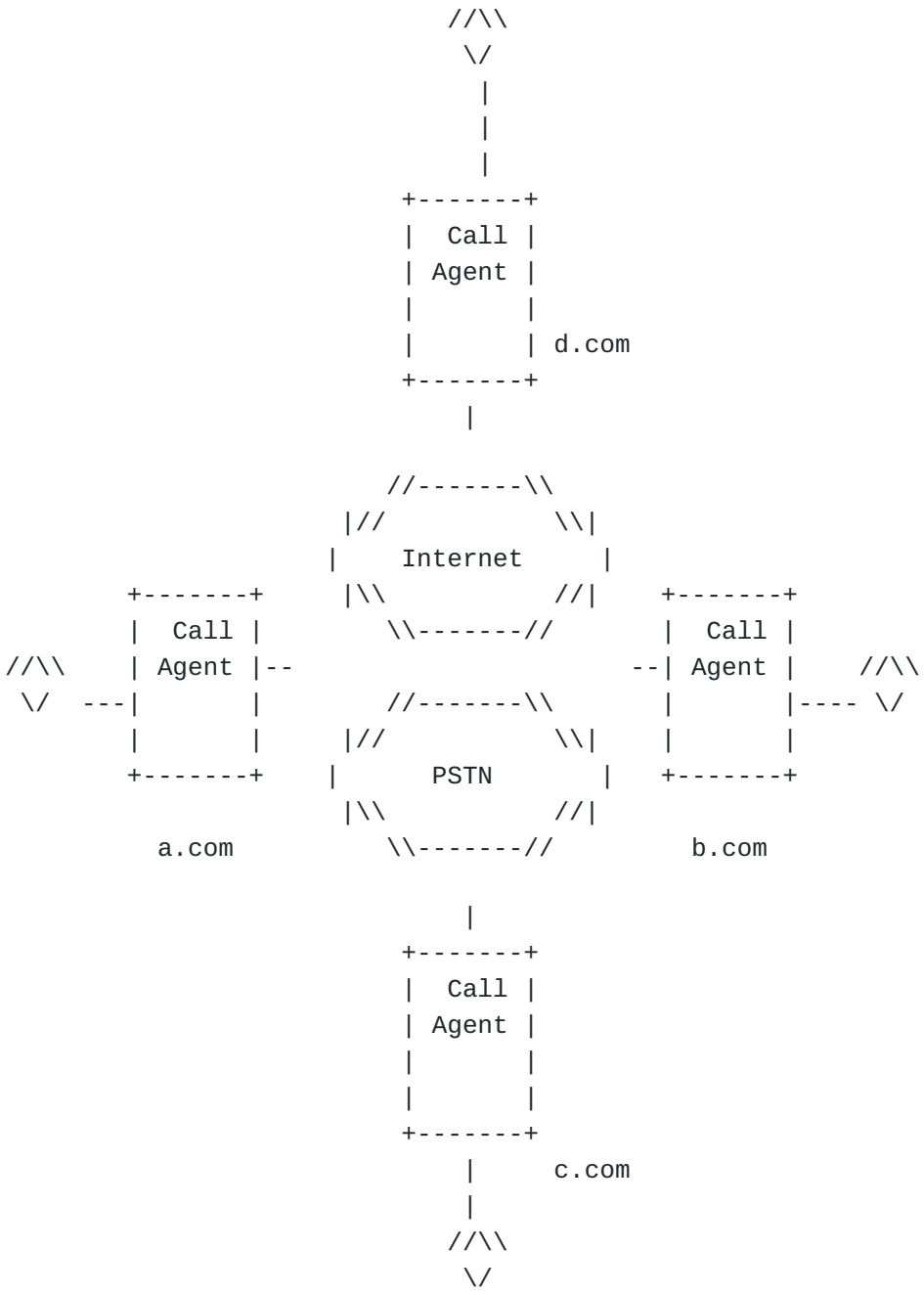

Figure 1: High Level Architecture

For purposes of explanation, it is easiest to think of each domain as having a single call agent which participates in the federation solution. In actuality, the functionality is decomposed into several sub-components, and this is discussed in more detail below. The call agent is connected to one or more phones in the domain, and is responsible for routing calls, handling features, and processing call

state. The call agent is stateful, and is aware of when calls start and stop.
Assume that all four domains have a 'fresh' installation of ViPR, and that domain b.com 'owns' +1 (408) 902-5xxx, a block of 1000 numbers allocated by its PSTN provider.
The ViPR mechanism can be broken into four basic steps: storage of phone numbers, PSTN first call, validation and caching, and SIP call.

5.3.1. Storage of Phone Numbers

TOC

The first step is that the call agents form a single, worldwide P2P network, using RELOAD [I-D.ietf-p2psip-base] (Jennings, C., Lowekamp, B., Rescorla, E., Baset, S., and H. Schulzrinne, “REsource LOcation And Discovery (RELOAD) Base Protocol,” March 2010.) with the Chord algorithm. This P2P network forms a distributed hash table (DHT) running amongst all participating domains. A distributed hash table is like a simple database, allowing storage of key-value pairs, and lookup of objects by key. Unlike a normal hash table, which resides in the memory of a single computer, a distributed hash table is spread across all of the servers which make up the P2P network. In this case, it is spread across all of the domains participating in the ViPR federation. The neat trick solved by Chord (and by other DHT algorithms), is an answer to the following: given that the desired operation is to read or write an object with key K, which node in the DHT is the box that currently stores the object with that key? Chord provides a clever algorithm which routes read and write operations through nodes in the DHT until they eventually arrive at the right place. With Chord, this will take no more than log2N hops, where N is the number of nodes in the DHT. Consequently, for a DHT with 1024 nodes, 10 hops are required in the worst case. For 2048, 11 hops. And so on. The logarithmic factor allows DHTs to achieve incredible scale and to provide enormous storage summed across all of the nodes that make up the DHT.
This logarithmic hopping behavior also means that each node in the DHT does not need to establish a TCP/TLS connection to every other node. Rather, connections are established to a smaller subset - just log(N) of the nodes.
In DHTs, each participating entity is identified by a node-ID. The node-ID is a 128 bit number, assigned randomly to each entity. They have no inherent semantic meaning; they are not like domain names or IP addresses.
In the case of ViPR, each call agent is identifed by one or more node-IDs. For purposes of discussion, consider the case where the call agent has just one. Each participating domain, including b.com in our example, uses the DHT to store a mapping from each phone number that it owns, to its own Node-ID. In the case of b.com, it would store 1000 entries into the DHT, each one being a mapping from one of its phone

numbers, to its own nodeID. Furthermore, when the mappings are stored, the mapping is actually from the SHA-1 hash of the phone number, to the nodeID of the call agent which claims ownership of that number. Pretending that the node-ID of the call agent in domain b.com is 0x1234 (a shorter 16 bit value to simplify discussion), the entries stored into the DHT by b.com would be:

Key	Value
SHA1(+14089025000)	0x1234
SHA1(+14089025001)	0x1234
SHA1(+14089025002)	0x1234
.....	
SHA1(+14089025999)	0x1234

Figure 2: DHT Contents

It is important to note that the DHT does not contain phone numbers (it contains hashes of them), nor does it contain IP addresses or domain names. Instead, it is a mapping from the hash of a phone number (in E. 164 format) to a node-ID.
b.com will store this mapping when it starts up, or when a new number is provisioned. The information is refreshed periodically by b.com. The actual server on which these mappings are stored depends on the Chord algorithm. Typically, the entries will be uniformly distributed amongst all of the call agents participating in the network.

5.3.2. PSTN First Call

TOC

At some point, a user (Alice) in a.com makes a call to +1 (408) 952-5432, which is her colleague Bob. Even though both sides have ViPR, the call takes place over the plain old PSTN. Alice talks to Bob for a bit, and they hang up.
At a random point of time after the call has completed, the call agent in a.com "wakes up" and says to itself, "that's interesting, someone in my domain called +1 (408) 952-5432, and it went over the PSTN. I wonder if that number is reachable over IP instead?". To make this determination, it hashes the called phone number, and looks it up in the DHT. It is important to note that this lookup is not at the time of an actual phone call - this lookup process happens outside of any phone call, and is a background process.

The query for +1 (408) 952-5432 will traverse the DHT, and eventually arrive at the node that is reponsible for storing the mapping for that number. Typically, that node will not be b.com, but rather one of the other nodes in the network (for example. c.com). In many cases, the called number will not find a matching mapping in the DHT. This happens when the number that was dialed is not owned by a domain participating in ViPR. When that happens, a.com takes no further action. Next time there is another call to the same number, it will repeat the process and check once more whether the dialed number is in the DHT.
In this case, there is a match in the DHT, and a.com learns the node-ID of b.com. It then proceeds to the validation step. It is also possible that there are multiple matches in the DHT. This can happen if another domain - d.com for example - also claims ownership of that number. When there are multiple matching results, a.com learns all of them, and performs the validation step with each.

5.3.3. Validation and Caching

TOC

Why not just store the domain in the DHT, instead of the node-ID? In that case, once a.com performed the lookup, it would immediately learn that the number maps to b.com, and could then make a direct SIP call next time.
The main reason this doesn't work is security. The information in the DHT is completely untrusted. There is nothing so far that enables a.com to know that b.com does, in fact, own the phone number in question. Indeed, if multiple domains make a claim on the number, it has no way to know which one (if any) actually owns it.
To address this critical problem, ViPR utilizes a technique called phone number validation. Phone number validation is the key concept in ViPR. The essential idea is that a.com will connect to the b.com server, by asking the DHT to form a connection to b.com's nodeID. Once connected, a.com demands proof of ownership of the phone number. This proof comes in the form of demonstrated knowledge of the previous PSTN call. When a call was placed from a.com to +1 (408) 952-5432, the details of that call - including its caller ID, start time, and stop time, create a form of shared secret - information that is only known to entities that participated in the call. Thus, to obtain proof that b.com really owns the number in question, a.com will demand a knowledge proof - that b.com is aware of the details of the call. The only way that b.com could know these details is if it had received the call, and the only way it could have received the call is if it owned the phone number.
There are a great many details required for this validation protocol to be secured. It needs to handle the fact that call start and stop times won't exactly match on both sides. It needs to deal with the fact that many calls start on the top of the hour. It needs to deal with the fact

that caller ID is not often delivered, and when it is delivered, is not reliable. It needs to deal with the fact that a.com may in fact be the attacker, trying to use the validation protocol to extract the shared secret from b.com. All of this is, in fact, handled by the protocol.
The protocol is based on the Secure Remote Password for TLS Authentication (SRP-TLS) [RFC5054] (Taylor, D., Wu, T., Mavrogiannopoulos, N., and T. Perrin, "Using the Secure Remote Password (SRP) Protocol for TLS Authentication," November 2007.), and is described more fully in [I-D.rosenberg-dispatch-vipr-pvp] (Rosenberg, J. and C. Jennings, "The Public Switched Telephone Network (PSTN) Validation Protocol (PVP)," November 2009.).
At the end of the validation process, both a.com and b.com have been able to ascertain that the other side did in fact participate in the previous PSTN call. At that point, a.com sends its domain name to b.com (this is described in more detail below), and b.com sends to a.com - all over a secured channel - a SIP URL to use for routing calls to this number, and a ticket. The ticket is is a cryptographic object, opaque to a.com, but used by b.com to allow incoming SIP calls. It is similar in concept to kerberos tickets - it is a grant of access. In this case, it is a grant of access for a.com to call +1 (408) 952-5432, and only +1 (408) 952-5432.
The a.com call agent receives the SIP URI and ticket, and stores both of them in an internal cache. This cache builds up slowly over time, containing the phone number, SIP URI, and ticket, for those numbers which are called by a.com and validated using ViPR. Because the cache entries are only built for numbers which have actually been called by users in the enterprise, the size of the cache self-scales. A call agent supporting only ten users will build up a cache proportional to the volume of numbers called by ten people, whereas a call agent supporting ten thousand users will build up a cache which is typically a thousand times larger.

5.3.4. SIP Call

TOC

At some point in the future, another call is made to +1 (408) 952-5432. The caller could be Alice, or it could be any other user attached to the same call agent. This time, the call agent notes that it has a cached route for the number in question, along with a SIP URI that can be used to reach that route. It also has a ticket.
The a.com call agent attempts to contact the SIP URI by establishing a TCP/TLS connection to the SIP URI it learned. If this connection cannot be made, it proceeds with the call over the PSTN. This ensures that, in the event of an Internet failure or server failure, the call can still proceed. Assuming the connection is established, the a.com call agent sends a traditional SIP INVITE to the terminating call agent, over this

newly formed secure connection. The SIP call setup request also contains the ticket, placed into a new SIP header in the message. When this call setup request arrives at the b.com call agent, it extracts the ticket from the new SIP header. This ticket is an object, opaque to a.com, that was previously generated by the b.com call agent. Figure 3 (Ticket Validation Step 1) illustrates how this ticket is generated and used.

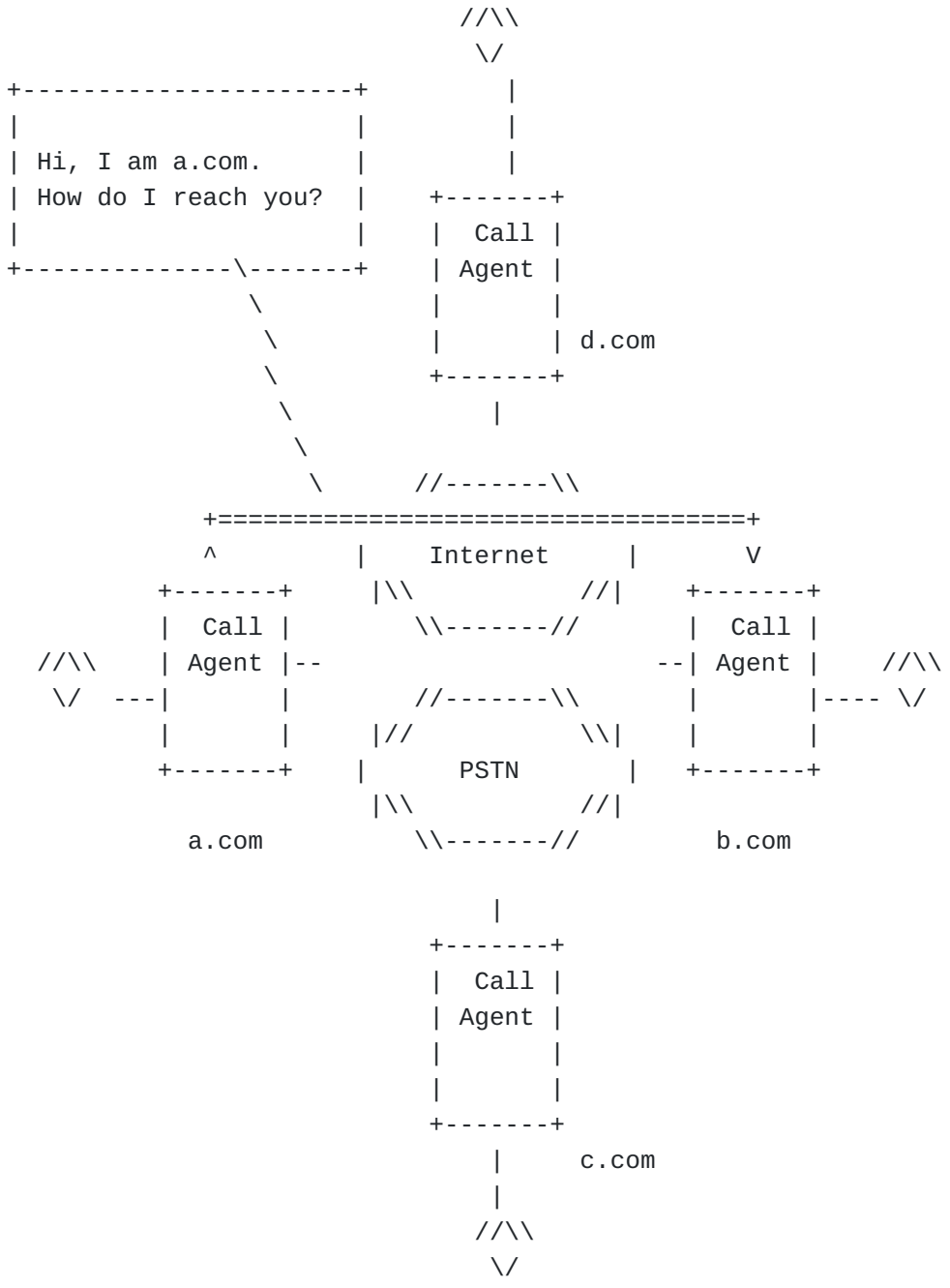

Figure 3: Ticket Validation Step 1

Towards the end of the validation process, domains a.com and b.com had determined that each was, in fact in possession of the shared secret information about the prior PSTN call. However, neither side has any information about the domain names of the other side. The originating domain - a.com - tells b.com that its domain name is a.com. It offers no proof of this assertion at this time.
Next, the b.com domain generates the ticket. The ticket has three fundamental parts to it:

1. The phone number that was just validated - in this case, +1 (408) 952-5432.

2. The domain name that the originating side claims it has - a.com in this case.

3. A signature generated by b.com, using a key known to itself only, over the other two pieces of information.

This ticket is then sent back to a.com at the end of the validation process, as shown in Figure 4 (Ticket Validation Step 2).

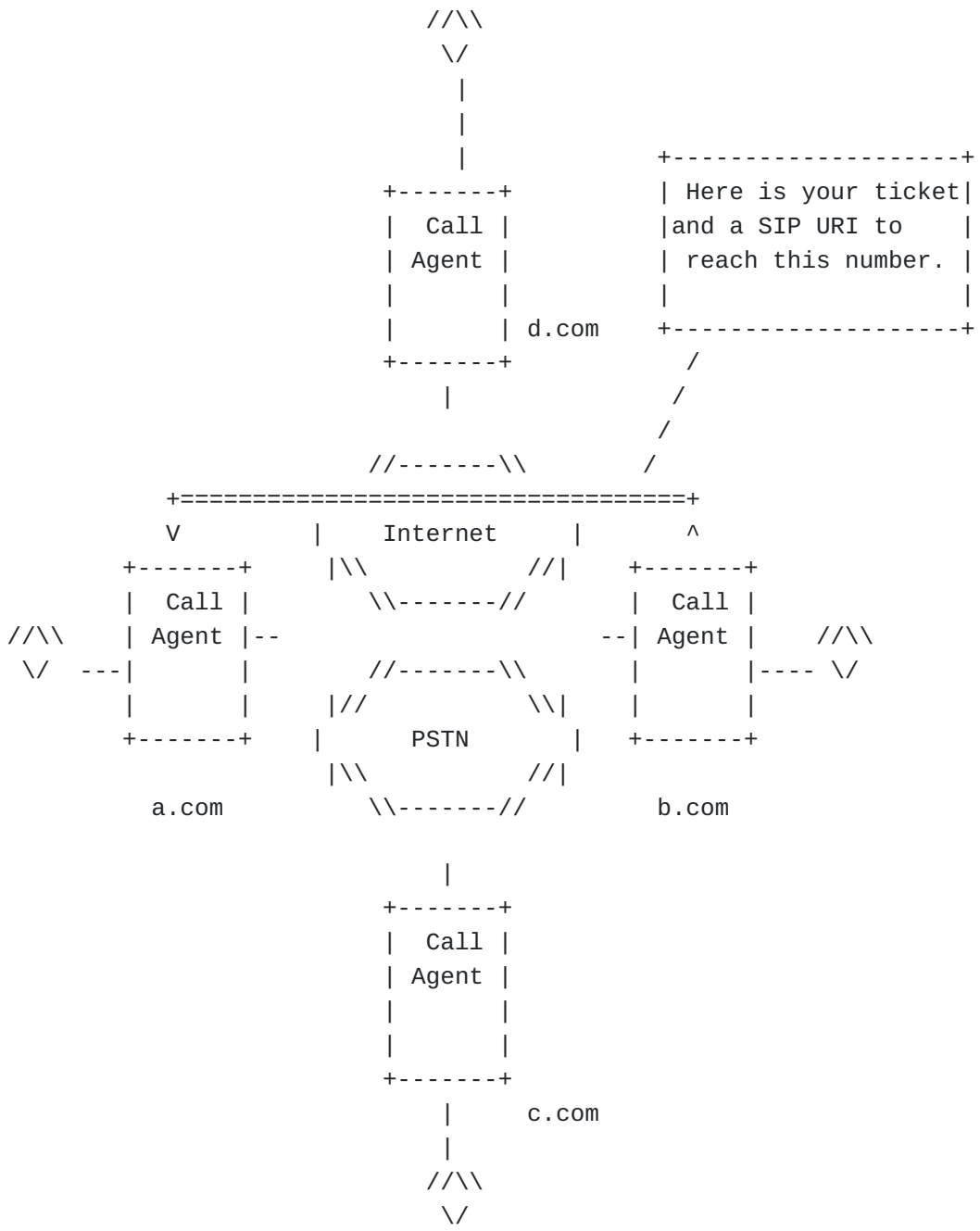

Figure 4: Ticket Validation Step 2

When a.com generates a SIP INVITE, it will contain this ticket. The INVITE arrives at the b.com call agent over the mutually authenticated TLS connection established between the domains.

The b.com call agent looks for the SIP header field in the INVITE that contains the ticket. First, it verifies the signature over the ticket. Remember that the b.com agent is the one that generated the ticket in

the first place; as such, it is in possession of the key required to validate the signature. Once validated, it performs two checks:

1. It compares the phone number in the call setup request (the Request URI) against the phone number stored in the ticket.

2. It compares the domain name of the calling domain, learned from the certificates in the mutual TLS exchange, against the domain name stored in the ticket.

If both match, the b.com call agent knows that the calling party is in fact the domain they claimed previously, and that they had in fact gone through the validation process successfully for the number in question. A consequence of this is that the following property is maintained:

> A domain can only call a specific number over SIP, if it had previously called that exact same number over the PSTN.

This property is key in fighting spam and denial-of-service attacks. Because calling numbers on the PSTN costs money - especially international calls - ViPR creates a financial disincentive for spammers. For a spammer to ring every phone in a domain with a SIP call, it must have previously called every number in the domain with a PSTN call, and had a successfully completed call to each and every one of them. Of course, once that PSTN call had been placed, the spammer would have already achieved their goals, and at cost. The additional VoIP call is not so exciting.
This property also means that, in order for an attacker to spam call numbers on VoIP, it must have already spam-called those same numbers on the PSTN. This means that the attacker would clearly be subject to regulations and laws governing usage of the PSTN for calling. As an example, a spammer in the United States would have already violated U.S. do-not-call rules by initiating the spam calls to the PSTN numbers.
It is important to note that ViPR does not completely address the spam problem. A large spamming clearing house organization could actually incur the costs of launching the PSTN calls to numbers, and then, in turn, act as a conduit allowing other spammers to launch their calls to those numbers for a fee. The clearinghouse would actually need to transit the signaling traffic (or, divulge the private keys to their domain name), which would incur some cost. As such, while this is not an impossible situation, the barrier is set reasonably high to start with - high enough that it is likely to deter spammers until it becomes a highly attractive target, at which point other mechanisms can be brought to bear. This is, again, an example of the incremental deployability philosophy that ViPR takes - let not the perfect be the enemy of the good.

---

6. Architecture Components and Functions

TOC

The architecture in Figure 1 (High Level Architecture) is overly simplistic. ViPR allows the functionality embedded within the call agent can be split up into three components, as shown in Figure 5 (Architecture):

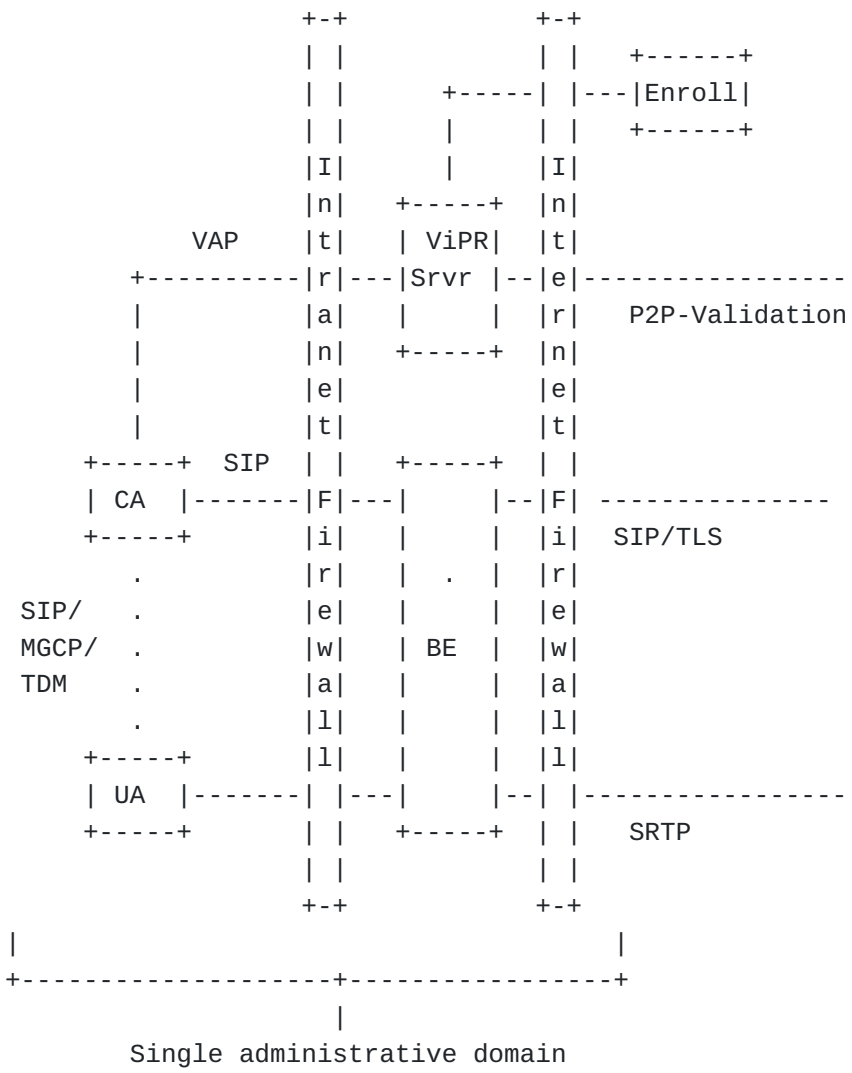

Figure 5: Architecture

Within each domain, there are three components that are ViPR-aware. These are the ViPR server, the call agent (CA), and the border element (BE). Outside of the domain, there is a P2P network and an enrollment server. A domain will typically have firewalls - an Internet firewall and an intranet firewall.

The sections which follow describe the roles and responsibilities of each component in more detail.

6.1. ViPR Server

TOC

The ViPR server is the heart of the system. It performs several key functions:

1. It implements the P2P protocol, acting as one or more nodes in the DHT. By placing this function separate from the call agent, it allows the call agent to be isolated from the traffic and security concerns that are often associated with a P2P network.

2. It implements the validation mechanism. It is informed of call events by the call agent, and sometime after the call, looks up the number in the DHT, and if found, attempts to connect to the node claiming ownership of the number, and then validates it.

3. It pushes newly learned routes to the call agent once validation has occurred. The ViPR server does not hold the call routes; this eliminates the need for an off-box query to perform call routing logic.

4. It stores numbers into the DHT. The call agent informs the ViPR servers of numbers to be published, and the ViPR server places them into the P2P network. Refreshing the stored numbers (by asking the ViPR server to restore them) is the responsibility of the call agent.

5. It implements a distributed quota enforcement algorithm, ensuring that malicious ViPR servers cannot store excessive data into the network.

6. It implements a policing function, pacing its store and fetch requests into the DHT to ensure that the network is not overwhelmed.

In order to join the P2P network and be able to receive incoming validation requests, the ViPR server must have open access to the public Internet. For this reason, it is typically placed into the DMZ. The Internet firewall will require two pinholes to be opened towards the ViPR server: one for the P2P protocol, and one for the validation protocol.
It is important to understand that the ViPR server does not perform any call processing. It does not process SIP or RTP traffic. It is a non-real-time server that performs validation processing in the background, outside of actual call attempts.

The ViPR server needs to connect with the call agent. This is done through the ViPR Access Protocol (VAP). VAP is described in more detail below.

6.2. Call Agent

TOC

The call agent is a box within the domain which performs call processing on behalf of one or more phones within the domain. ViPR can work with a wide variety of call agents, as long as they meet some specific criteria:

- The call agent must be know of the start time, stop time, caller ID, and called numbers of calls placed from phones towards the PSTN.

- The call agent must be capable of making routing decisions for outbound calls from phones that would otherwise go to the PSTN, directing them towards the PSTN or towards other domains (based on ViPR routing rules).

Based on this definition, many different types of products typically found within a domain could act as the call agent. An IP PBX or TDM PBX with a SIP interface can be the call agent. A Session Border Controller (SBC) that connects calls from a PBX to the PSTN, can act as the call agent. An IMS application server can act as the call agent. A PSTN gateway, used for all calls egressing a domain from a set of phones, can act as a call agent.
A SIP proxy can act as a call agent; as long as it is capable of stashing the relevant call information into Record-Route headers for usage at the end of the call, it can even operate without retaining call state.
A single phone can also act as the call agent, representing itself and its own phone number.
In ViPR, the call agent performs several key functions specific to ViPR:

- It informs the ViPR server of the phone numbers to be stored in the DHT for its domain.

- It refreshes those numbers in the DHT, redoing the storage operation periodically.

- At the end of a call, the call agent sends a ViPR Call Record (VCR) to the ViPR server, containing the start time, stop time, caller ID and called party number.

*It learns validated routes from the ViPR server. These routes consist of a phone number, a SIP URI to utilize when contacting that phone number, and a corresponding ticket. The call agent is responsible for storing those routes.

*When a call is to be made towards a PSTN number, the call agent is responsible for checking whether there is a route for that number, learned via a prior notification from the ViPR server. If so, it is responsible for sending the INVITE towards the learned SIP URI, and for including the ticket the X-Cisco-ViPR-Ticket header field.

Those functions which require communications with the ViPR server are done by implementing VAP. VAP is a client-server protocol, with the call agent acting as the client, and the ViPR server acting as the server. For this reason, the call agent is sometimes called the VAP client or ViPR client.

6.3. Border Element

TOC

The border element is responsible for the SIP layer perimeter security functions. In particular:

*The border element ensures that all egress SIP traffic is carried over TLS. Border elements must reject any incoming SIP requests which are not over TLS. SIP over TLS is mandatory-to-use in ViPR, and it must be performed using mutual TLS.

*The border element ensures that all egress RTP traffic is actually carried using SRTP. If the traffic originated by the UA in the domain is inherently SRTP, the criteria is met. However, many domains do not utilize SRTP internally, and if it is not used internally, the border element must convert to SRTP. Similarly, the border element is responsible for rejecting any incoming SIP calls that are not set up with SRTP. SRTP is mandatory in ViPR.

*The border element ensures that ingress and egress SIP traffic is 'fixed up' so that it can pass through the Internet firewall succesfully. Typically, this is done using a traditional SBC/ALG function.

*The border element inspects all incoming SIP INVITEs, and performs ticket verification. In this process, it looks for the X-Cisco-ViPR-Ticket header field in the INVITE. If not present, it discards the request. If present, it verifies the signature, and then compares the called number and remote TLS domain against

the contents of the ticket. If they do not match, the border element discards the INVITE.

The border element can perform other, non-ViPR tasks, as is common for border elements. These include header inspection and validation, anti-virus checks on embedded content, SIP state machine conformance, policy checks on various services, and so on.
The role of the border element can be fulfilled by any number of products typically found within domains. These include Session Border Controllers and firewalls. Indeed, the border element function can be embedded directly in the Internet firewall.
The border element is connected to the call agent via SIP, and to the user agent (UA) via RTP. The border element has no direct connection to the ViPR server. However, in order for ticket processing to work in this model, the ViPR server and border element must share a secret that is used to create the tickets. This is discussed in more detail below.

6.4. Enrollment Server

TOC

P2P protocols - including RELOAD - require the usage of an enrollment server in order to obtain the certificates that are used to secure the network. ViPR uses, and indeed requires, that all RELOAD traffic be over TCP/TLS with mutual authentication. The certificates used are obtained through an enrollment process. The details on how P2P enrollment are done are beyond the scope of this document.

6.5. P2P Network

TOC

The collection of ViPR servers form a single, worldwide, P2P network utiizing RELOAD and the Chord algorithm.
It is very important to understand that the DHT is never accessed in real-time. It is not queried at call setup time. This is because the DHT is slow, involving many hops. Queries could take seconds.
Furthermore, we don't want to rely on proper operation of the DHT to actually make calls.

---

7. Protocols

TOC

The overall ViPR solution utilizes several protocols, each peforming a different function.

7.1. P2P: RELOAD

TOC

ViPR utilizes the RELOAD protocol [I-D.ietf-p2psip-base] (Jennings, C., Lowekamp, B., Rescorla, E., Baset, S., and H. Schulzrinne, "REsource LOcation And Discovery (RELOAD) Base Protocol," March 2010.) to run amongst each of the ViPR servers. Each ViPR server acts as one or more nodes in the DHT. The number of nodes that the ViPR server implements directly determines the quota allocated to that ViPR server, and in turn, the amount of work it must perform storing data.
ViPR, however, does not implement the SIP usage that has been defined for RELOAD [I-D.ietf-p2psip-sip] (Jennings, C., Lowekamp, B., Rescorla, E., Baset, S., and H. Schulzrinne, "A SIP Usage for RELOAD," March 2010.). That is because the DHT is not used as a traditional distributed registrar. Instead, it implements a new usage - the ViPR usage - which stores phone numbers. It also utilizes the DHT for storage of certificates, using a certificate usage.

7.1.1. ViPR Usage

TOC

The ViPR usage is described in detail in [I-D.rosenberg-dispatch-vipr-reload-usage] (Rosenberg, J. and C. Jennings, "A Usage of Resource Location and Discovery (RELOAD) for Public Switched Telephone Network (PSTN) Verification," November 2009.). This section provides a brief overview.
The ViPR usage makes use of the dictionary type. Each resource ID is a key, computed by taking the SHA1 hash of an E.164 formatted phone number. The value stored at this resource-ID is a dictionary. The dictionary entries are the set of virtual ViPR servers which claim ownership of those numbers.
Since a ViPR server might support a multiplicity of call agents from different domains, it is necessary to logically segment a ViPR server so that - from a security perspetive - it operates logically like different virtual ViPR servers, one for each call agent. Each virtual instance of a ViPR server is called a VService. Thus, the entries in the dictionary are key value pairs whose key is the concatenation of the node-ID and an identifier for the VService within that node. The value at each key is the node-ID to contact for validation.
When a node in the DHT receives a Store request, and it is the responsible node for the resource ID, it will verify that the node-ID in both the key and value of the dictionary entry match the node-ID in the certificate it presents. This ensures that one ViPR server can never overwrite data from another ViPR server.
The ViPR usage also specifies a quota mechanism. Unlike the SIP usage, where there are very specific rules about what resource-IDs a node may store into the DHT, with ViPR, there is no way to restrict what

resource IDs may be stored by a ViPR server. This is because, in ViPR, the resource IDs are derived from phone numbers, and at the time of storage, there is no way to know whether the node performing the store actually owns this phone number. Consequently, a responsible node will accept stores from any node for any resource ID. However, to limit malicious users from consuming all of the resources of the DHT, the ViPR usage imposes a quota on storage. Each node performing a store is allocated a fixed quota on the number of records it can place into the DHT. A probabilistic enforcement model is utilized at each responsible node based on the fraction of the hashspace owned by that responsible node. Roughly speaking, if the system quota is 10,000 phone numbers per node-ID, if a responsible node owns 10% of the DHT, it will accept an average of 1000 phone numbers from any one single node-ID.

7.1.2. Certificate Usage

TOC

Further details pending.

7.2. ViPR Access Protocol (VAP)

TOC

The ViPR Access Protocol (VAP) is documented in [I-D.draft-rosenberg-dispatch-vipr-vap] (Rosenberg, J. and C. Jennings, “Validation Access Protocol (VAP),” November 2009.).
VAP is a client-server protocol that runs between the call agent and the ViPR server. VAP is a simple, binary based, request/response protocol. It utilizes the same syntactic structure and transaction state machinery as STUN [RFC5389] (Rosenberg, J., Mahy, R., Matthews, P., and D. Wing, “Session Traversal Utilities for NAT (STUN),” October 2008.), but otherwise is totally distinct from it. VAP clients initiate TCP/TLS connections towards the ViPR server. The ViPR server never opens connections towards the call agent. This allows the ViPR servers to run on the public side of NATs and firewalls.
Once the connections are established, the call agent sends a Register message to the ViPR server. This register message primarily provides authentication and connects the client to the ViPR server. VAP provides several messages for different purposes:

*Publish: The Publish message informs the ViPR server of service information. There are two types of Publishes supported in ViPR. The first is the ViPR Service (VService). This informs the ViPR server of the SIP URIs on the call agent and black and white lists used by the ViPR server to block validations. The ViPR server stores that information locally and uses it during the validation process, as described above. The second Publish is the

ViPR number service. The ViPR server, upon receiving this message, performs a Store operation into the DHT.

*UploadVCR: This message comes in two flavors - an originating and terminating message. An originating UploadVCR comes from a call agent upon completion of a non-VIPR call to the PSTN. A terminating UploadVCR comes from an agent upon completion of a call received FROM the PSTN. The ViPR server behavior for both messages is very different. For Originating UploadVCR, the ViPR server will store these, and at a random time later, query the DHT for the called number and attempt validation against the ViPR servers that are found. For a terminating UploadVCR, the ViPR server will store these, awaiting receipt of a validation against them.

*Subscribe: Call agents can subscribe for information from the VipR server. There is one service that UCM can subscribe for: number Service. When a new number is validated, the ViPR server will send a Notify to the call agent, containing the validated number, the ticket, and a set of SIP trunk URIs.

*Notify: The ViPR server sends this message to the call agent when it has an event to report for a particular subscription.

The VAP protocol provides authentication by including an integrity object in each message. This integrity message is the hash of the contents of the message and a shared secret between the ViPR server and the client. VAP can also be run over TLS, which enhances security further.
The P2P network introduces rate limits for the purposes of performance management and limiting denial of service attacks. Each node in the DHT comes with it a limit on the amount of stores per second, reads per second, and total amount of data it can store in the DHT. The ViPR server rigorously follows those limits.
As a consequence, when numbers are stored into the DHT, they are written in slowly based on the rate limits. The call agent will send a Publish operation for each individual number. The ViPR server will perform the store in a rate-limited fashion. When the store is complete, the ViPR server responds to the Publish, and the call agent can move to the next DID to publish. Thus, it may take hours or even days to fully store the set of numbers into the DHT. The process then repeats several days later in order to refresh the data in the DHT.

7.3. Validation Protocol

TOC

The core of ViPR is the validation protocol. The validation protocol is used by one ViPR server to connect to another, demand proof-of-

knowledge of a previous PSTN call, and once proven, securely learn a SIP URI and ticket for usage in future SIP calls between domains.
The validation protocol is documented in [I-D.rosenberg-dispatch-vipr-pvp] (Rosenberg, J. and C. Jennings, “The Public Switched Telephone Network (PSTN) Validation Protocol (PVP),” November 2009.).
The validation protocol is built using TLS-SRP [RFC5054] (Taylor, D., Wu, T., Mavrogiannopoulos, N., and T. Perrin, “Using the Secure Remote Password (SRP) Protocol for TLS Authentication,” November 2007.). TLS-SRP creates a secure TLS connection, but instead of using certificates, utilizes a password. TLS-SRP was designed for cases where the passwords are relatively weak. In the case of the validation protocol, the passwords are formed from parameters of a previous PSTN call. Once a secure TLS connection is formed, a simple request/response protocol is run over it. The request contains the domain name of the originating ViPR server, and the response contains the SIP URI and ticket for that number.
The validation protocol properly handles time offsets between the two domains for the start and stop times of the calls, the relatively weak entropy of a single phone call, the grand chessmaster attack, and non-delivery or inaccurate delivery of caller-ID, amongst other issues. The validation protocol can be tuned by administrators to allow for arbitrary levels of security, measured in terms of equivalent entropy. The equivalent entropy is the number of bits of entropy that must be demonstrated, as if the domains were authenticating each other using a password with that amount of entropy. This gives domains a 'nerd knob' they can turn to trade off security for performance.
Because the validation protocol utilizes TLS-SRP, it does not run directly through the DHT. This is why a ViPR server requires a separate pinhole to be opened for the validation protocol.

7.4. SIP Extensions

TOC

The connection between the call agents in different domains is SIP. ViPR requires that the inter-domain connections run over TLS, and furthermore, utilize SRTP keyed with Sdescriptions.
ViPR extends SIP with its anti-spam mechanism. This takes the form of a ticket, present in a SIP header field.
[I-D.rosenberg-dispatch-vipr-sip-antispam] (Rosenberg, J. and C. Jennings, “Session Initiation Protocol (SIP) Extensions for Blocking VoIP Spam Using PSTN Validation,” November 2009.) defines this header field and the format of the ticket it contains.

TOC

8. Example Call Flows

This section provides call flows for the key use cases.

8.1. PSTN Call and VCR Upload

TOC

A call flow for the initial PSTN call and VCR upload is shown in Figure 6 (PSTN Call and Upload).

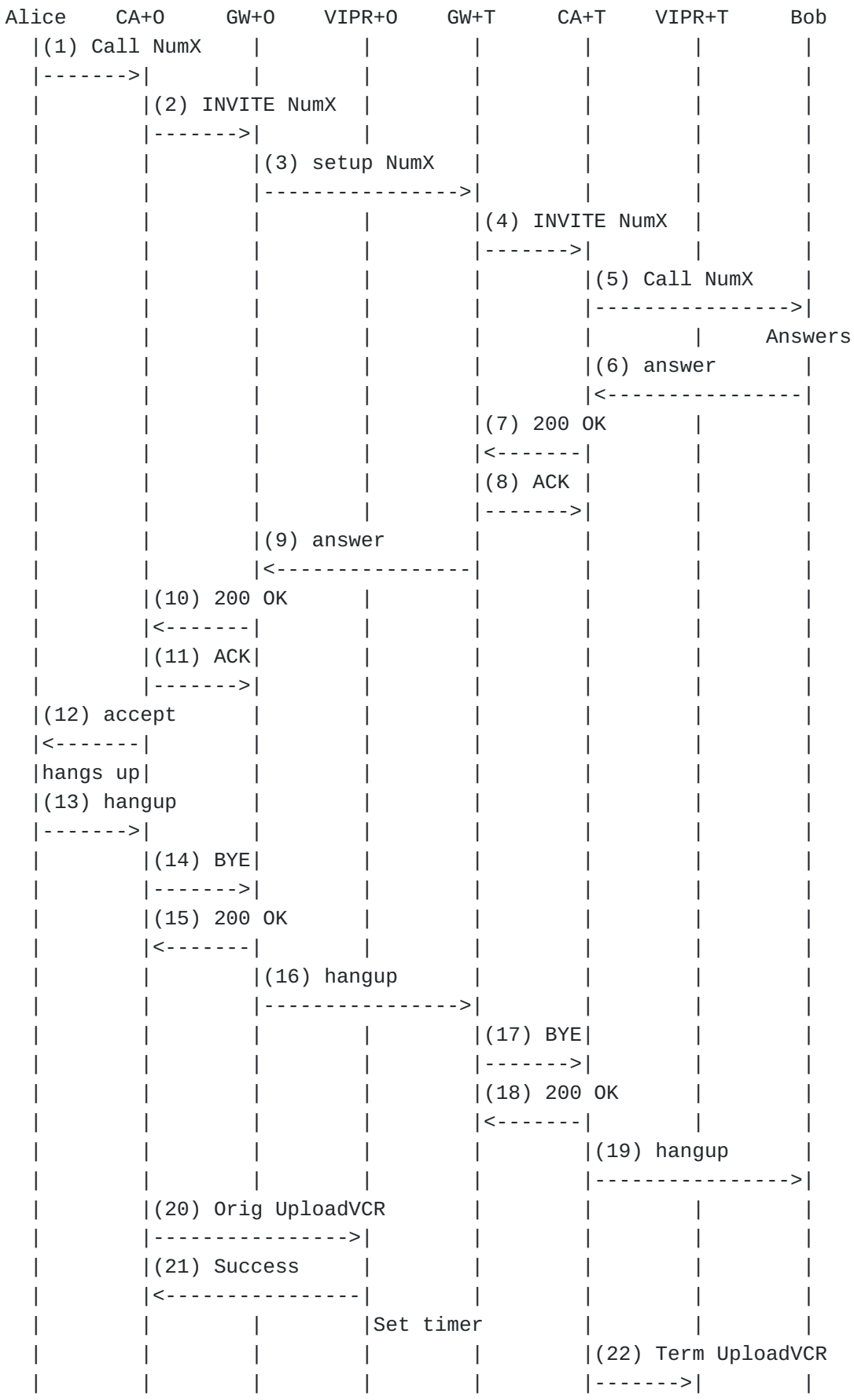
Alice
CA+O
GW+O
VIPR+O
GW+T
CA+T
VIPR+T
Bob
(1) Call NumX
(2) INVITE NumX
(3) setup NumX
(4) INVITE NumX
(5) Call NumX
Answers
(6) answer
(7) 200 OK
(8) ACK
(9) answer
(10) 200 OK
(11) ACK
(12) accept
hangs up
(13) hangup
(14) BYE
(15) 200 OK
(16) hangup
(17) BYE
(18) 200 OK
(19) hangup
(20) Orig UploadVCR
(21) Success
Set timer
(22) Term UploadVCR

Figure 6: PSTN Call and Upload

In message 1, Alice calls the number of her colleauge, Bob. This is NuMX. This call is routed over the PSTN, through the terminating call agent, and rings Bob's phone (messages 1-5). Bob answer the phone, and this is propagated back to Alice (messages 6-12). Bob and Alice talk for a while, and then Alice hangs up. This hangup is propagated to Bob, and the call is terminated (messages 13-19).
The originating call agent notes that this call went to the PSTN, and might be a candidate for a future SIP call. It sends an UploadVCR message to its ViPR server (message 20), containing the start time, stop time, callerID and called party number. The ViPR server acknowledges this (message 21), and then sets a timer for a random time into the future, at which point it will attempt validation. The terminating side is similar; it sends an UploadVCR to its ViPR server (message 22), which is acknowledged (message 23). The terminating side does not set a timer; it waits for a possible validation attempt which may or may not arrive in the future.

8.2. DHT Query and Validation

TOC

This section provides the call flow for what happens on the originating ViPR server when the timer fires, in Figure 7 (Validation Flow).

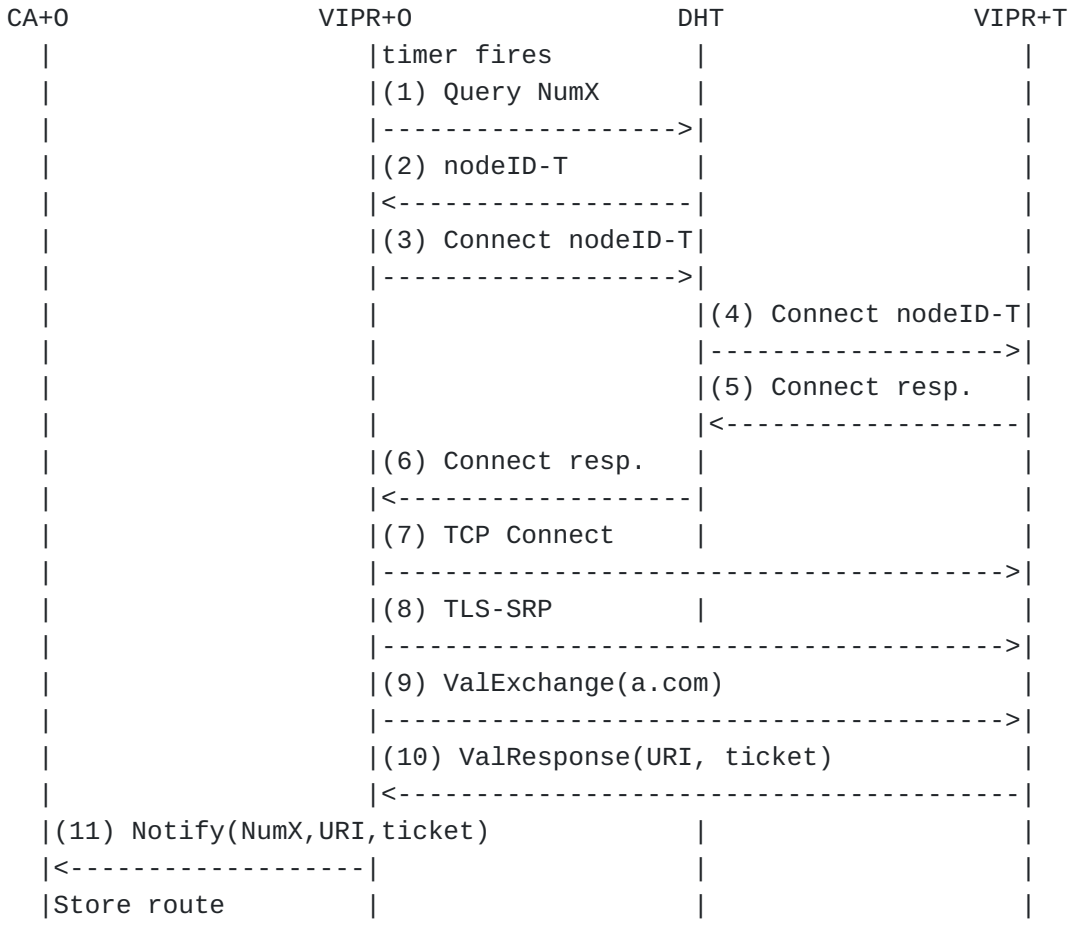

Figure 7: Validation Flow

First, the timer that was set by the originating ViPR server in Figure 6 (PSTN Call and Upload) fires. When it fires, the ViPR server examines the called party number from the VCR. It performs a query into the DHT, to see if this number has been stored by any domain (message 1). In this case, it has, and the DHT returns with a successful query response (message 2). This response indicates that the terminating ViPR server, with nodeID T, claims ownership of the number.
The originating ViPR server asks the DHT to form a connection between itself and the terminating ViPR server. This message exchanges IP addresses and ports through which a TCP connection can be attempted; details are omitted (messages 3-6). Now, the originating ViPR server can establish a TCP connection to the terminating ViPR server (message 7). Next, the originating ViPR server begins negotiation of a TLS-SRP connection. The TLS-SRP uses the caller ID and called number as a "username" for this exchange, and the start time and stop time of the call as a password. As both sides share the same values for this

secret, the secure connection is established. This is now a TLS connection between the two ViPR servers.
Over this secure connection, the originating ViPR server sends a ValExchange request. This request contains the domain name that is claimed by the originating ViPR server (this claim is not verified at this time) (message 9). This is received by the terminating ViPR server, which then creates a ticket for that domain and numX, and passes the ticket and the SIP URI back to the originating VipR server (message 10). The originating VipR server sends this information to its call agent (message 11), which then stores it for usage in a future call.

8.3. DHT Query and No Match

TOC

In this case, after the PSTN call of Figure 6 (PSTN Call and Upload), the timer fires, but the originating ViPR server finds no match in the DHT. This is an alternative case to the flow in Figure 7 (Validation Flow).

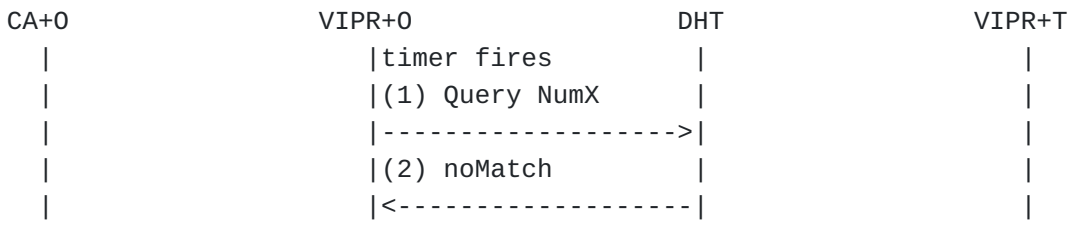

Figure 8: DHT No-Match

8.4. SIP Call

TOC

In this case, shown in Figure 9 (SIP Call), a user makes a call to a number which has been learned via ViPR.

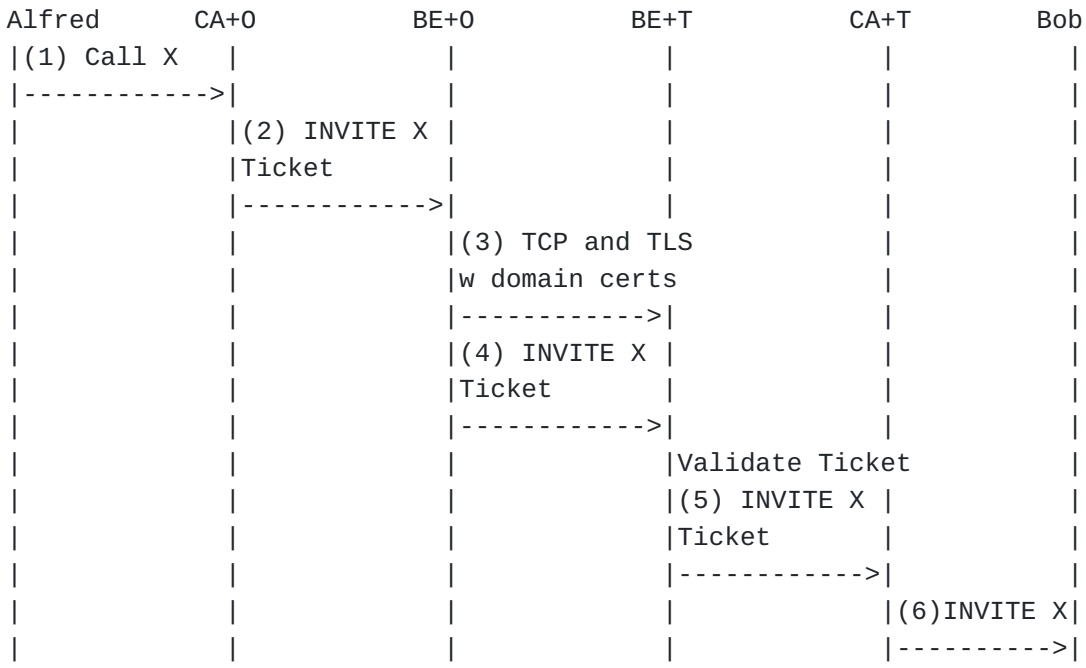

Figure 9: SIP Call

First, a user in the originating domain - Alfred - calls Bob's number (message 1). The originating call agent notes that it has a cached route for that number. It extracts the SIP URI, using it as the topmost Route header field, and then attaches the ticket to the X-Cisco-ViPR-Ticket header field. This INVITE is sent to a default next hop border element (message 2). The border element establishes a TCP/TLS connection with the domain in the Route header. It uses a traditional domain certification for this TLS connection (message 3). Once established, it sends the INVITE over the connection (message 4). This arrives at the terminating call agent, which extracts the ticket and verifies it. To verify it, it checks the signature using the key that was used to create the ticket. Then, it compares the domain name in the ticket with the domain name from the TLS connection handshake. Finally, it compares the called party number in the Request-URI with the value from the ticket. Assuming they all match, the call is forwarded to the terminating call agent (message 5), where it is finally delivered to Bob (message 6).

---

9. Security Considerations

TOC

Security is incredibly important for ViPR. This section provides an overview of some of the key threats and how they are handled.

9.1. Attacks on the DHT

TOC

Attackers could attempt to disrupt service through a variety of attacks on the DHT.
Firstly, it must be noted that the DHT is never used at call setup time. It is accessed as a background task, solely to learn NEW numbers and routes that are not already known. If, by some tragedy, an attacker destroyed the P2P network completely, it would not cause a single call to fail. Furthermore, it would not cause calls to revert to the PSTN - calls to routes learned previously would still go over the IP network. The only impact to such a devastating attack, is that a domain could not learn *new* routes to new numbers, until the DHT is restored to service. This service failure is hard for users and administrators to even notice.
That said, VIPR prevents many of these attacks. The DHT itself is secured using TLS - its usage is mandatory. Quota mechanisms are put into place that prevent an attacker from storign large amounts of data in the DHT. Other attacks are prevented by mechanisms defined by RELOAD itself, and are not ViPR specific.

9.2. Theft of Phone Numbers

TOC

The key security threat that ViPR is trying to address is the theft of phone numbers. In particular, a malicious domain could store, into the DHT, phone numbers that it does not own, in an attempt to steal calls targeted to those numbers. This attack is prevented by the core validation mechanism, which performs a proof of knowledge check to verify ownership of numbers.
An attacker could try to claim numbers it doesn't own, which are claimed legitimately by other domains in the ViPR network. This attack is prevented as well. Each domain storing information into the DHT can never overwrite information stored by another domain. As a consequence, if two domains claim the same number, two records are stored in the DHT. An originating domain will validate against both, and only one will validate - the real owner.
An attacker could actually own a phone number, use it for a while, validate with it, and build up a cache of routes at other domains. Then, it gives back the phone number to the PSTN provider, who allocates it to someone else. However, the attacker still claims ownership of the number, even though they no longer have it. This attack is prevented by expiring the learned routes after a while. Typically, operators do not re-assign a number for a few months, to allow out-of-service messages to be played to people that still have the old number. Thus, the TTL for cached routes is set to match the duration that carriers typically hold numbers.

An attacker could advertise a lot of numbers, most of which are correct, some of which are not. ViPR prevents this by requiring each number to be validated individually.
An attacker could make a call so they know the call details of the call they made and use this to forge a validation for that call. They could then try to convince other users, which would have to be in the same domain as the attacker, to trust this validation. This is mitigated by not sharing validations inside of domains where the users that can originate call from that domain are not trusted by the domain.

9.3. Spam

TOC

Another serious concern is that attackers may try to launch VoIP spam (also known as SPIT) calls into a domain. ViPR prevents this by requiring that a domain make a PSTN call to a number before it will allow a SIP call to be accepted to that same number. This provides a financial disincentive to spammers. The current relatively high cost of international calling, and the presence of national do-not-call regulations, have prevented spam on the PSTN to a large degree. ViPR applies those same protections to SIP connections.
As noted above, ViPR still lowers the cost of communications, but it does so by amortizing that savings over a large number of calls. The costs of communications remain high for infrequent calls to many numbers, and become low for frequent calls to a smaller set of numbers. Since the former is more interesting to spammers, ViPR gears its cost incentives away from the spammers, and towards domains which collaborate frequently.
Of course, ViPR's built-in mechanism is not a guarantee. A SPIT clearinghouse could shoulder the costs of the PSTN calls, and then re-sell its access for a fee. However, this still causes the clearinghouse to utilize non-trivial resources in its attack. Though these costs are less than the PSTN, they are more than zero, and should act as a deterrent for a long while.

9.4. Eavesdropping

TOC

Another class of attacks involves outsiders attempting to listen in on the calls that run over the Internet, or obtain information about the call through observation of signaling.
All of these attacks are prevented by requiring the usage of SIP over TLS and SRTP. These are mandatory to use.

---

10. IANA Considerations

TOC

This specification does not require any actions from IANA.

---

11. References

TOC

11.1. Normative References

TOC

[I-D.ietf-p2psip-base]	Jennings, C., Lowekamp, B., Rescorla, E., Baset, S., and H. Schulzrinne, “REsource LOcation And Discovery (RELOAD) Base Protocol,” draft-ietf-p2psip-base-08 (work in progress), March 2010 (TXT).
[I-D.ietf-p2psip-sip]	Jennings, C., Lowekamp, B., Rescorla, E., Baset, S., and H. Schulzrinne, “A SIP Usage for RELOAD,” draft-ietf-p2psip-sip-04 (work in progress), March 2010 (TXT).
[I-D.rosenberg-dispatch-vipr-reload-usage]	Rosenberg, J. and C. Jennings, “A Usage of Resource Location and Discovery (RELOAD) for Public Switched Telephone Network (PSTN) Verification,” November 2009.
[I-D.rosenberg-dispatch-vipr-sip-antispam]	Rosenberg, J. and C. Jennings, “Session Initiation Protocol (SIP) Extensions for Blocking VoIP Spam Using PSTN Validation,” November 2009.
[I-D.draft-rosenberg-dispatch-vipr-vap]	Rosenberg, J. and C. Jennings, “Validation Access Protocol (VAP),” November 2009.
[I-D.rosenberg-dispatch-vipr-pvp]	Rosenberg, J. and C. Jennings, “The Public Switched Telephone Network (PSTN) Validation Protocol (PVP),” November 2009.

11.2. Informative References

TOC

[RFC2543]	Handley, M., Schulzrinne, H., Schooler, E., and J. Rosenberg, “SIP: Session Initiation Protocol,” RFC 2543, March 1999 (TXT).
[RFC3261]	Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., Handley, M., and E. Schooler, “SIP: Session Initiation Protocol,” RFC 3261, June 2002 (TXT).

[RFC3263]	Rosenberg, J. and H. Schulzrinne, “Session Initiation Protocol (SIP): Locating SIP Servers,” RFC 3263, June 2002 (TXT).
[RFC5039]	Rosenberg, J. and C. Jennings, “The Session Initiation Protocol (SIP) and Spam,” RFC 5039, January 2008 (TXT).
[RFC3761]	Faltstrom, P. and M. Mealling, “The E.164 to Uniform Resource Identifiers (URI) Dynamic Delegation Discovery System (DDDS) Application (ENUM),” RFC 3761, April 2004 (TXT).
[RFC5389]	Rosenberg, J., Mahy, R., Matthews, P., and D. Wing, “Session Traversal Utilities for NAT (STUN),” RFC 5389, October 2008 (TXT).
[RFC5067]	Lind, S. and P. Pfautz, “Infrastructure ENUM Requirements,” RFC 5067, November 2007 (TXT).
[RFC5054]	Taylor, D., Wu, T., Mavrogiannopoulos, N., and T. Perrin, “Using the Secure Remote Password (SRP) Protocol for TLS Authentication,” RFC 5054, November 2007 (TXT).
[I-D.roach-sipping-clf-syntax]	Roach, A., “Binary Syntax for SIP Common Log Format,” draft-roach-sipping-clf-syntax-01 (work in progress), May 2009 (TXT).
[I-D.kaplan-sip-session-id]	Kaplan, H., “A Session Identifier for the Session Initiation Protocol (SIP),” draft-kaplan-sip-session-id-02 (work in progress), March 2009 (TXT).

Authors' Addresses

TOC

	Jonathan Rosenberg
	jdrosen.net
	Monmouth, NJ
	US
Email:	jdrosen@jdrosen.net
URI:	http://www.jdrosen.net
	Cullen Jennings
	Cisco
	170 West Tasman Drive
	MS: SJC-21/2
	San Jose, CA 95134
	USA
Phone:	+1 408 421-9990
Email:	fluffy@cisco.com