An Architecture for Peer-to-Peer Session Initiation Protocol (P2P SIP)
draft-shim-sipping-p2p-arch-00

Status of this Memo

By submitting this Internet-Draft, each author represents that any applicable patent or other IPR claims of which he or she is aware have been or will be disclosed, and any of which he or she becomes aware will be disclosed, in accordance with Section 6 of BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

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."

The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt.

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html.

This Internet-Draft will expire on August 30, 2006.

Copyright Notice

Copyright (C) The Internet Society (2006).

Abstract

This document describes an architecture for peer-to-peer SIP systems in which a separate and independent peer-to-peer overlay layer provides a distributed resource placement and search service for SIP. It also aims to help identify what should be specified for interoperation.
# Table of Contents

1. Introduction ............................................. 3
   1.1 Scope of the architecture document ................. 3
2. Terminology .............................................. 4
3. Architecture Overview .................................... 5
4. SIP entity operations ..................................... 8
   4.1 P2P UA Behavior ....................................... 9
   4.2 P2P Proxy Behavior ................................... 9
   4.3 P2P Registrar Behavior ............................... 10
   4.4 Example Call Flows ................................. 11
       4.4.1 Call Flow between P2P UAs ..................... 11
       4.4.2 Call Flow between CS UA and P2P UA within the same domain ............... 12
5. Peer Initiation .......................................... 14
   5.1 Bootstrapping ......................................... 14
   5.2 Association and Authentication ....................... 16
   5.3 Association ........................................... 16
   5.4 Authentication ........................................ 17
   5.5 NAT/FW Traversal ..................................... 17
6. Registration ................................................ 18
7. Becoming a Super Node ..................................... 18
8. Formats of Resource Records for SIP ...................... 19
9. P2P Overlay API .......................................... 20
10. What Needs To Be Specified .............................. 22
11. Security Considerations .................................. 23
   11.1 Bootstrapping Security ............................... 23
   11.2 ON/SN Authentication ................................ 23
   11.3 Peer Transport Security ............................. 23
   11.4 Firewalls ............................................. 24
   11.5 Network Address Translators ......................... 24
   11.6 Registration .......................................... 24
   11.7 Session Endpoint Discovery and Security ............. 25
   11.8 Ill Behavior of Ordinary Nodes ....................... 25
   11.9 Ill Behavior of Super Nodes ......................... 26
12. Acknowledgements ......................................... 26
13. References ................................................ 26
   13.1 Normative References ................................. 26
   13.2 Informative References ................................ 27
Authors’ Addresses .......................................... 27
Intelectual Property and Copyright Statements ............... 29
1. Introduction

The Session Initiation Protocol (SIP) [1] has been used largely in the client/server model. In the client/server model, the User Agent nodes, being client nodes, typically use services provided by dedicated server nodes such as SIP Proxy, SIP Registrar, and location server. Those servers are physically different from the nodes with User Agents, statically configured to play their roles, and managed by administrators.

In comparison, in the Peer-to-peer (P2P) networking model, the participating nodes, or peers, which are sharing their computing and networking resources, perform all or some of the server functions in a distributed fashion, thus eliminating need for all or some of the dedicated servers. There are two fundamental functions required to realize the P2P networking model. First, the peers, distributed randomly over the physical network, should be able to form an overlay network, which is managed in an automatic and distributed fashion. This function is called herein the P2P overlay management function that mainly handles peers joining and leaving the overlay network. Second, the peers should be able to discover resources that are distributed among the peers, and furthermore, should be able to place resources among the peers when remote peers dynamically create the resources for use. This function is called the P2P Service Function that mainly handles lookup and placement of resources. The above two functions together are referred to as ‘the P2P overlay functions’.

SIP can be used in the P2P networking model. In particular, the P2P service function can be used to provide a distributed location service for SIP. For convenience of presentation, SIP operating in the P2P networking model is called ‘P2P SIP (Peer-to-Peer Session Initiation Protocol)’.

1.1 Scope of the architecture document

This document aims to describe an architecture for systems based on P2P SIP. The key characteristic of the P2P SIP architecture is that a separate layer, independent of SIP, provides the P2P overlay functions and SIP is an application using the services of the layer. In this document, the components of the P2P overlay network are identified along with the specification of the overall network architecture of the P2P overlay. This document also defines new P2P SIP logical entities and their behavior corresponding to the logical entities (UA, Proxy, Registrar) defined by the SIP specification. Example call procedures are presented to demonstrate how these new P2P SIP entities will operate to establish SIP signaling. The initiation procedure required for a new peer to become part of the P2P overlay is also specified. This version of the document assumes
all the peers in the network are well-behaved nodes and thus ignores the possible denial of service problems due to nodes not contributing to the well being of the overlay, in the main text. An initial discussion on this is provided in the Security Considerations Section 11. This is an important problem that will be addressed in future versions of the document.

2. Terminology

In this document, words which are normally key words, such as "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" are used colloquially and are not intended to be interpreted as described in RFC2119 [1].

Peer-to-Peer (P2P) Architecture: An architecture in which nodes (peers) cooperate together to perform tasks. Each node has essentially equal importance and performs the same tasks within the network. Additionally, nodes communicate directly with one another to perform tasks. Occasionally, nodes with better resources (such as not being behind NATs) may have a more significant role.

(P2P Overlay) Bootstrapping: Finding a peer already participating in the target P2P overlay network in order to join the P2P overlay network.

Bootstrap seed(s): A super node on the P2P overlay network that is used as the first point of contact for new P2P entities coming online.

Bootstrap server(s): A server on the network that provides a peer with the address of a node already in the P2P overlay network.

Neighbor or Neighboring Peer: A peer whose contact information such as IP address is known and maintained by the local peer.

P2P Overlay Entity: An entity in the protocol stack that implements the P2P overlay functions.

P2P Overlay Functions: The P2P overlay management function and the P2P service function.

P2P Overlay Layer: A protocol layer that handles the P2P overlay functions.

P2P Overlay Management Function: Peer initiation and handling join and leave of peers to maintain a P2P overlay network.

P2P Proxy: A SIP proxy on a device with the P2P overlay layer.

P2P Registrar: A SIP registrar on a device with the P2P overlay layer.

P2P Service Function: Placing resources on certain peers and discovering locations of target resources in the P2P overlay network, in short, placement and lookup of resources.
P2P SIP System: A system in which P2P SIP is used to realize applications such as VoIP, Presence and Instant Messaging.
P2P SIP: SIP running over a P2P overlay network in which SIP resources, such as user location information, are managed by a P2P overlay network.
P2P UA (P2P User Agent): A SIP UA (user agent) on a device with the P2P overlay layer.
Peer Initiation: Bootstrapping and any other tasks necessary for a peer to reach the state in which the peer can perform placement and search in the P2P overlay network.
Peer: A network node offering its own resources for other peers and using resources of other peers for its own good. The term ‘peer’ is used interchangeably with ‘node’ in the context of peer-to-peer overlay networks.
Peer-to-Peer (P2P) Overlay Network: An overlay network formed and self-managed by the participating peers.
Resource Record: A record about a resource that may include the resource itself or include just meta-data about the resource. To share a resource, a peer creates a resource record that is accessible to other peers. Looking up a resource is equivalent to looking up a resource record.
Distributed Hash Table (DHT): A mechanism in which resources are given a unique key produced by hashing some attribute of the resource, locating them in a hash space (see below). Nodes located in this hash space also have a unique identifier within the hash space. Nodes store information about resources with keys that are numerically similar to the node’s identifier in the hash space [13].

Terminology defined in RFC3261 [2] is used without definition.

3. Architecture Overview

A peer in the P2P SIP system is comprised of two layers: the P2P overlay layer and the SIP layer. The P2P overlay layer handles all the P2P overlay functions. The P2P overlay layer does not interpret semantic meanings of the resources placed or looked up by the higher layers including SIP. But its design goal is serving SIP and thus SIP-based real-time media applications. That is, other applications may use the P2P overlay layer presented here but the architecture and the algorithm used for this P2P overlay layer may not necessarily be optimal for all applications.

From the perspective of SIP, the P2P overlay layer provides mainly a location service for various SIP resources. One of the most important information, or resource, for SIP operation is the user location, i.e., the IP address corresponding to a SIP URI [2]. Typically, the caller precisely knows the SIP URI of the call...
recipient before a call is made. DHT (Distributed Hash Table) based structured P2P networks support efficient search mechanisms when the resource names are precisely known, thus, meets the requirements of P2P SIP well. Therefore, we propose the use of DHT for the P2P overlay layer.

P2P overlay networks can be comprised of peers with different amount of physical resources operating in various network environments. The larger a P2P overlay network is, the more significant the heterogeneity of peers will be. Even though every peer is ideally supposed to contribute its physical resources to the services of the P2P overlay network for other peers, peers without proper physical resources had better be exempted from the obligation for better performance of the P2P overlay network in overall.

So peers are classified into two types: super nodes (SNs) and ordinary nodes (ONs). For DHT-based P2P overlay networks, super nodes participate in building and maintaining the DHT. Ordinary nodes do not participate in DHT. Super nodes serve search requests by routing the search traffic or generating replies to search requests. To discover a resource from other peers, an ordinary node sends a search request message to a super node. Then the super node routes the search request message in the DHT. In a sense, super nodes comprise a distributed storage and search service network. Either an ordinary node or a super node could use the service provided by the overlay network. Depending on the system requirements, some systems based on P2P SIP may be comprised of only super nodes. A farm of SIP proxies using P2P SIP among themselves is an example of such a system. In this document, unless particularly mentioned, a system is assumed to have the two types of peers. The P2P SIP specification will contain descriptions of the protocols between the ON-SN pair and the SN-SN pair.

Since only the super nodes participate in the DHT, the P2P overlay network has a hierarchical structure: the core consisting of super nodes forming a DHT with a periphery of ordinary nodes.

Each ordinary node is associated with one or more super nodes. The relationship between the ordinary node and those super nodes is like that of a client and servers or of a host and routers. The ordinary node uses the storage and lookup services of the overlay network core through the associated super nodes. SIP has its own hierarchical structure. SIP proxies form a kind of infrastructure providing call request routing services and SIP UAs use the services. SIP event servers store the event subscription records and distribute event information to the subscribers. In general, the hierarchy of peers in the P2P overlay layer is independent of the hierarchical structure of SIP. A UA with P2P overlay layer is referred to as P2P UA in this
document, and similarly a proxy or a registrar with P2P overlay layer is referred to P2P Proxy or P2P-Registrar respectively. For example, it is possible to build a P2P VoIP network comprised of only P2P UAs running over a P2P overlay network with the distinction of super nodes and ordinary nodes. It is also possible to run a P2P Proxy process over an ordinary node. Super nodes are good candidates to be P2P-Proxy but not all super nodes in a P2P overlay network have to become a P2P Proxy. The choice of SIP entity on a peer node will depend on the specific system design. Authorization to join the P2P overlay network is based only on user credentials, i.e., a device is authorized based on the credential of the user or the service using the device. A user can change devices freely and use multiple devices simultaneously. Authentication and authorization is provided primarily by a central logical entity named a login server. How a user subscribes to a login server is out of scope of this document. Out-of-band methods such as web pages can be used.

There are four types of entities in the P2P overlay network, a logical view of which is depicted in figure 1.

```
+---------+    +------------+       +----+
| Login   |    | Bootstrap  |       | ON |      +----+
| Server  |    | Server     |       +----+      | ON |
+---------+    +------------+        |          +----+

+----+                                  V           |
| ON |--------> +----+              +----+ <--------+
+----+ --+      | SN | ============ | SN |          +----+

Figure 1: A logical view of the P2P overlay network with four types of entities: Ordinary Node, Super Node, Login Server, and Bootstrap Server.

Once a node knows one or multiple nodes participating in a P2P overlay network, it can start the process to join the network. A
### bootstrap server

A bootstrap server is an entity that provides a new node with the information about existing nodes in the target P2P overlay network.

The login server and the bootstrap server facilitate formation of a P2P overlay network. When other means become available for their functions, they can be omitted. In this sense, they are optional entities.

### 4. SIP entity operations

The separation of the P2P overlay layer and the SIP layers allows for the possibility of a P2P SIP network that consists of multiple P2P overlay networks with a P2P Proxy routing SIP signaling messages from one overlay to another. These individual P2P overlay networks MAY either correspond to a single SIP domain where every P2P SIP entity on the particular overlay network will be from the same SIP domain, or a P2P overlay network MAY have P2P SIP entities from multiple SIP domains. When a P2P UA tries to establish a call to another P2P UA in the same P2P overlay network, the INVITE message will be sent directly to the destination P2P UA based on a lookup in the P2P overlay network. If a P2P UA tries to establish a call to another P2P UA in a different P2P overlay, the INVITE message will be forwarded to a P2P Proxy of the local overlay network which in turn will route the INVITE message to the final destination through the a P2P Proxy associated with the remote domain.

The P2P service function may be used for SIP in various forms, resulting in different network composition scenarios. There are three key logical entities defined by SIP [SIP], a User Agent (UA), a Proxy and a Registrar. Each of these SIP entities could potentially include a P2P overlay layer and become part of the P2P overlay network. These SIP entities with a P2P overlay layer are referred to as P2P UA, P2P Proxy or P2P-Registrar. Different combination of the P2P and Client-Server (CS) SIP entities could be combined to create the network composition of a particular SIP domain. Table 1 summarizes what we consider as meaningful network composition scenarios and the roles of the P2P layer from the SIP perspective.

<table>
<thead>
<tr>
<th></th>
<th>UA</th>
<th>Proxy</th>
<th>Registrar</th>
<th>Role of P2P Overlay Layer</th>
</tr>
</thead>
<tbody>
<tr>
<td>(a)</td>
<td>P2P</td>
<td>P2P</td>
<td>Not</td>
<td>Replace the Registrar, the location database and DNS lookup for local proxy</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Required</td>
<td></td>
</tr>
<tr>
<td>(b)</td>
<td>CS</td>
<td>P2P</td>
<td>P2P</td>
<td>Replace the location database</td>
</tr>
</tbody>
</table>
Table 1. Network composition scenarios for a SIP domain

In the above table, 'P2P' implies that the corresponding SIP entity is on a peer device participating in the P2P overlay and 'CS' implies a client-server SIP entity. The behavior of each P2P SIP entity is described below.

4.1 P2P UA Behavior

The P2P UA uses the P2P service function to store its user location information in the P2P overlay network or to look up user location information for target UAs it may want to contact. So, for user registration, the P2P UA creates a user location record as described in Section 8 and requests its local P2P overlay layer to place the user location record in the overlay network by calling the P2P overlay API functions, either add() or update(). Based on the lifetime set in the user location record, the P2P UA periodically refreshes the user location record. The configuration information required by the P2P UA is one or more of the SIP-URIs by which it can be contacted and the identifier of the overlay network which the P2P UA should join.

To make an outgoing call, a P2P UA SHOULD try retrieving the resource record in the following order:
- The target UA associated with the callee.
- The local Proxy associated with its own domain.

It is important to note that these location records should be verifiable by means of some secure signature based on credentials derived from the login server. If both of these lookups fail, the P2P UA MAY try a DNS lookup for the local proxy associated with its own domain. A P2P UA MAY be configured to try the local proxy always. The default behavior described above SHOULD be followed in the absence of such a configured choice.

Once the next hop is identified and an INVITE message is sent to that node, the P2P UA behaves the same way as the UA in the client server model. For incoming calls, the P2P UA behaves the same way as the UA in the client server model.

4.2 P2P Proxy Behavior

A P2P Proxy plays the role of a gateway between two P2P overlay networks or between a client-server SIP domain and a P2P overlay network. Hence, a P2P Proxy MAY be a part of more than one P2P overlay network. Once the local P2P overlay layer completes initiation and joins the P2P overlay network, a P2P proxy generates its location record based on its domain name and requests its local
P2P overlay layer entity to place the record in the overlay network. Since there can be multiple proxies for the same domain, the P2P proxy uses add() function of the P2P overlay API to place its location record. The P2P Proxy is responsible for keeping the location record on the overlay up-to-date by adding a new record as the old record gets close to expiration. The configuration information required by P2P Proxy is the domain name it is associated with and the overlay identifier that corresponds to the particular domain.

Upon receiving an INVITE message, a P2P proxy determines whether the callee’s URI belongs to a domain it supports or a remote domain. In the former case, the P2P proxy MUST lookup user location records of the callee in the corresponding overlay network and forwards the INVITE message to the location of the callee. Once having found the location or multiple possible locations of the callee and verifying the authenticity of these location record(s), the P2P proxy behaves the same way as the proxy in the client server model. If the lookup in the P2P overlay fails and it knows a SIP location server for the local domain, the P2P proxy MAY contact the SIP location server for the callee’s location.

- If the callee’s URI belongs to a remote domain, the P2P proxy SHOULD locate the remote proxy associated with the URI using DNS as specified by the client-server SIP specification.

The architecture as described in this document allows for the possibility of constructing a transit P2P overlay network among P2P Proxies that can be used to locate the P2P Proxy of the remote domain. In order to achieve this, the overlay identifier corresponding to the transit network must be configured into these P2P Proxies. In such a scenario, the P2P Proxy will try a lookup of the remote proxy on this overlay network before falling back to DNS lookup as a backup to reach other proxies that are not part of this transit overlay network.

4.3 P2P Registrar Behavior

When a P2P overlay network is used to store location information of CS UAs (UAs operating in the client server model), the Registrar serving the CS UAs is configured to participate in the P2P overlay network, thus called a P2P Registrar. If a P2P Registrar is configured, it functions as the front of the P2P overlay network. When a SIP registration message is received, the P2P Registrar generates a user location record for the registering UA and requests the P2P overlay layer to place the location record in the overlay network.
4.4 Example Call Flows

The following example call flows provide an indication of the messages required to establish sessions in various environments. Further work is required in this area, and the following is presented for indicative purposes only.

4.4.1 Call Flow between P2P UAs

Now let’s look at how a simple one-to-one call is established between two P2P UAs. Figure 2 illustrates an example sequence of message transfer during call establishment where both the caller and the callee are P2P UAs and have public IP addresses. Also, both P2P UAs are in the same overlay network. It is assumed that there is no firewall blocking SIP or the P2P overlay protocol between peers. In this scenario, the user on node 1 (an ON) is the caller and the user on node 4 (an ON) is the callee. The user location record of the callee happens to be stored at node 3. With a call establishment command from the application, the UAC is given the SIP URI of the callee. The UAC does not have the location information for the callee, thus generates the resource key for the user location of the callee and requests the P2P overlay layer to search for the callee’s location by calling the P2P overlay API described in Section 9 (step 1).

Upon receiving the search request, the P2P overlay layer generates a search request message that contains the resource key from the SIP layer and sends it to the associated super node (step 2). The associated super node checks its local storage and forwards the search request message to the next hop according to the DHT routing table (step 3) if it does not have the target resource. The peer who has the target resource sends a search reply message containing the resource, i.e., the callee’s user location record in this case (step 4). The search reply message is directly sent back to the SN associated with the caller peer, i.e. the search reply does not retrace the path of the search message. Upon receiving the search reply message, the associated super node forwards the reply to the calling ordinary node (step 5). The P2P overlay layer of the calling peer returns the resource to the SIP layer (step 6). The UAC interprets the user location record and identifies the IP address and the port number for the UAS. Then the UAC generates an INVITE message and sends it to the UAS directly (step 7). From this point, the two UAs can communicate directly with each other.
4.4.2 Call Flow between CS UA and P2P UA within the same domain

It is possible to run CS UAs and P2P UAs within the same domain. Such a composition of a network is useful when CS UAs were already deployed and P2P UAs are incrementally deployed.

The SIP Registrar is configured to operate as P2P Registrar. The CS UAs sends registration messages to the P2P Registrar. Then the P2P Registrar places the location information of CS UAs in the overlay network via the P2P overlay layer. The P2P UAs place their location information in the overlay network themselves via the P2P overlay layer. Figure 3 illustrates an example sequence of message transfer during call establishment where the caller is a CS SIP UA and the callee is a P2P UA. In this scenario, both have public IP addresses without firewall restrictions on SIP or overlay messages. Note that the overlay network is represented by word ‘overlay’ and messages between peers in the P2P overlay layer are not depicted.

UA1, a CS UA, sends an INVITE message, aiming to establish a call with UA2, to its proxy, which happens to be connected to the P2P overlay network (Step 1). This proxy then resolves location of UA2.
In this case, the proxy queries the P2P overlay network (Step 2). The reply contains UA2’s location record (step 3). Subsequently, the INVITE message is forwarded to the discovered location (Step 4). Upon receiving the INVITE message, UA2 sends a reply message to the proxy (step 5) that, in turn, forwards it to UA1.

```
Figure 3. A call from a CS UA to a P2P UA
```

When UA2, a P2P UA, calls UA1, a CS UA, UA2 can discover UA1’s location from the P2P overlay network (step 1 and 2). After that, UA2 sends an INVITE message to UA1 directly (step 3). In this case, there is no need to go via a Proxy. Figure 4 illustrates this call procedure.
5. Peer Initiation

When a node decides to participate in the peer-to-peer SIP network, it needs to connect to other nodes, and identify peer super nodes (SN). In order to do this, it starts operation as an ordinary node (ON). After identifying if it is able to contact SN that compose the DHT, a peer may become a SN itself.

Below is a summary of the sequence of functions undertaken by a node in joining the peer-to-peer network. This peer initiation must be done independently for each P2P overlay network it is trying to join. Subsections follow describing all of these operations:

1. Bootstrap
2. Association and authentication.
3. NAT/FW traversal (At this stage, outbound sessions are possible).

In order to receive inbound calls, the peer will then have to register at least one SIP URI as described in Section 6, and later the peer MAY also want to become a SN as described in Section 7. Once a node is a SN, it agrees to service requests by peer SNs, and allow ONs to connect, register and search through it.

5.1 Bootstrapping

A node wishing to join a peer-to-peer overlay network needs to select
which overlay to join. While a peer may join different overlays, it will be necessary to maintain separate authorization, registration, hash tables, as well as separate lists of bootstrap servers. When discerning which network to join, a node may select to bootstrap only SNs which have one particular overlay identifier, or to use a default identifier. This overlay identifier needs to be exchanged in peer-to-peer messages, as each peer may itself be participating in multiple P2P overlay networks.

Bootstrapping the ON requires identification of SNs whom a peer should try to contact initially. Connecting to these SNs provides a mechanism whereby the peer can contact the peer-to-peer network, establish trust within that network and start participating in sessions.

Addresses for the initial contact may be gained by methods outlined below.

1. Service location (multicast)
2. Cached addresses
3. Last good address
4. Pre-configured bootstrap server(s).

Ordering of attempts to contact SNs is needed for a number of reasons. We RECOMMEND the above order, as described:

(1) Service Location

A peer can try sending a local multicast request to find if there are other P2P SIP nodes in the same local area network or administrative domain.

When P2P SIP is used within a corporate network, this solution for bootstrapping will be able to identify local resources and even in the global scenario, once a single node in the same broadcast/multicast domain joins the P2P overlay, other nodes can bootstrap locally.

As an example, peers may be able use the Service Location Protocol (SLP) to identify SNs participating in the P2P network [3][4]. Where a directory agent is not configured on the network, peers multicast queries to find a SN, which would act as an SLP Service Agent. If an existing directory agent is found, this may provide a list of active SNs.

(2) Cached Addresses:

A peer may retain information about SNs proposed by its SN during its
prior connection to the network. The list allows the SN to notify the ON of dynamic changes in service availability, and shows if the ON should first try to use the same SN again.

(3) Last Good Address:

A peer may attempt to contact its last known SN, with which it had a successful connection. This address may be tried after learned cached addresses, in order to allow for robustness and spreading of ONs between SNs within the network.

(4) Preconfigured Bootstrap Server:

The IP address or domain name of a server in the Internet that maintains a list of currently available SN addresses could be configured in peers. This server may not participate in the peer-to-peer network, and the protocol to access the server list may be an existing protocol like HTTP.

This address is contacted last, as the bootstrap servers are considered potential failure points, and the intention is to reduce the load on these devices.

5.2 Association and Authentication

After gathering information about potential SNs, the peer should contact candidates, and attempt to authenticate with one or more of them.

5.3 Association

Selection of the transport for contacting the SN is provided through the bootstrap process. As such, where multiple choices exist for a particular IP address or the information about the SN does not specify which protocol and port to contact the SN on, the ON must select amongst the following mechanisms by which to contact the peer.

By default, it may be useful to contact the server in the following order:
- UDP
- TCP
- Fallback Transport.

UDP is favored, as the SN may need to retain less state for each associated ON.

The Fallback Transport may be HTTP based, and potentially proxied, in order to gain access through networks which otherwise would not be
able to make these lookups.

Initial contact should confirm the presence of the SN, and ensure that the SN is functioning in that role. Additional handshakes to perform mutual return reachability tests may also be incorporated.

5.4 Authentication

In order to participate in the peer-to-peer network as an ordinary node, credentials MAY be needed to show a node’s right to retrieve and update information. After the association

Additionally, the ON will wish to authenticate the SN, in order to identify that the connection between the SN and ON is direct, and that the SN has sufficient authority to undertake operations on the ON’s behalf in the P2P network.

An example authentication scheme is to make use of TLS, in order to authenticate the ON and SN and derive a secured channel for communicating between them [6]. Where UDP based operations are undertaken, the datagram-based version of TLS is used analogously [draft-rescorla-dtls]. In this scenario, each node requires a certificate from a mutually acceptable source, and SNs are guaranteed to already possess one as they would have gone through the initiation procedure themselves.

5.5 NAT/FW Traversal

In order to undertake peer-to-peer communications with other nodes on the Internet, SIP messages will need to be exchanged, and media channels set up between end nodes. In order to ensure communications are available at session initiation time, it is necessary to test whether an ON is behind a NAT or a restrictive firewall, which may limit session establishment from that address.

Therefore, peers should test reachability, for example, by performing ICE operations, with the SN as the STUN server in order to identify what type of connectivity they possess [11][5]. Additional configuration of relay servers may be necessary, and the SN can identify a TURN server to contact (which may be itself or another SN). Upon completion of this testing, a set of viable derived addresses SHOULD be generated on the ON, for use in session initiation [10].

An ON MAY immediately start retrieving peers’ URIs and initiating outbound calls.
6. Registration

In order to receive inbound calls, a peer needs to publish information in the p2p network listing addresses and ports using which it may be contacted. Using the reachability information gained through NAT and Firewall traversal operations, a peer may advertise a set of addresses and ports by instructing the SN to place or replace records in the DHT referring to the ON’s own identity. Multiple records may be managed by specifying the preference relationships as described in ICE [11]. The API for managing data on the P2P overlay is presented in Section 9.

7. Becoming a Super Node

Super nodes are self-selected dynamically and automatically. In order to become a SN, a node

1. MUST be able to receive messages on predetermined protocols and ports from other SNs on the overlay network.
2. SHOULD be online stably. - This requirement may need a metric to be defined in the future to make it more specific.
3. SHOULD have sufficient physical resources such as CPU power, memory size, and network bandwidth. - This requirement may need a set of metrics to be defined in the future to make it more specific.

If a peer wishes to become a SN, it MUST be able to receive P2P overlay management and service function messages from other peers on pre-defined protocols and ports. When a P2P overlay network spans over the global Internet and contains peers private IP addresses as well as peers with public IP addresses, SNs need to perform STUN, and possibly TURN server operations thus it is RECOMMENDED that SNs should be on the public Internet, and publicly addressed [5][10]. If all the peers of a P2P overlay network are in the same address realm, for example, a LAN behind a NAT, SNs don’t have to have, or, actually can’t have public IP addresses.

SN operation requires that the peer reserve space to store a portion of the hash table, and transfer data into and out of its own storage, as it or other devices enter or leave the set of SNs. If SNs stay online only for a short period, signaling traffic to restore the DHT becomes heavy and search performance may degrade significantly. So it is desirable to select nodes that are likely to stay online longer. Different from the first requirement, the second and third requirements provide only relative criteria. Specific threshold values for the metrics of the two requirements should be decided depending on the target system characteristics.
8. Formats of Resource Records for SIP

The peer placing a resource and the peer retrieving the resource via search should use the same format for the resource record and the same algorithm to assign a key to a given resource record. Otherwise, the resource cannot be discovered or interpreted properly. The resource record is specific to the application that creates and uses the resource record. It is opaque to the P2P overlay layer.

An example of a user location record is given below.

```
<resource>
  <version>1.0</version>
  <!---------------- resource format version -->
  <type>user location</type>
  <!------------------ type of the resource -->
  <key>19873761ab24</key>
  <!---------------- key for the resource -->
  <lifetime>3600</lifetime>
  <!---------------- lifetime of the record -->
  <timestamp>19809832142</timestamp>
  <!---------------- indicate which is more recent -->
  <user_URI>user@example.com</user_URI>
  <location>
    <node_IP>178.14.234.21</node_IP>
    <!------------------ the IP address at which the user can be reached -->
    <transport>TCP5060 UDP5060 TCP80 TCP443</transport>
    <!---------------- the list of ports the UA is listening to -->
  </location>
  <location>
    <node_IP>192.168.0.100</node_IP>
    <!------------------ the IP address at which the user can be reached -->
    <transport>TCP5060 UDP5060 TCP80 TCP443</transport>
    <!---------------- the list of ports the UA is listening to -->
  </location>
</resource>
```

The algorithm for resource key generation depends on the type of resource. In general, it has the following form.

\[
\text{key} = \text{hash}(\text{resource name}).
\]

The resource name is composed as a concatenation of the resource type name and a string identifying the resource instance within the type. For user location information, the user URI can identify the resource instance, and, therefore, the key is generated as follows:

\[
\text{key} = \text{hash}("\text{user location"+<user URI>}),
\]

where "user location"+<user_URI> is a concatenation of "user location" and the value of user_URI element. For the above example
user location, the input for the key generation function is "user location|user@example.com". Using the concatenation method, the user URI can be used to identify many types of resources. For the hash function, a cryptographic hash function like SHA-1 [12] should be used in order to distribute keys uniformly in the key space.

Another type of record is the proxy location record. An example of a proxy location record is given below.

```xml
<resource>
  <version>1.0</version>
  <!-------- resource format version-->
  <type>proxy location</type>
  <!-------- type of the resource-->
  <key>19873761ab24&l;/key>
  <!-------- key for the resource-->
  <lifetime>36000</lifetime>
  <!-------- lifetime of the record-->
  <timestamp>198023422</timestamp>
  <!-------- indicate which is more recent-->
  <domain>example.com</domain>
  <location>
    <node_IP>178.14.234.21</node_IP>
    <!-------- the IP address at which the user can be reached-->
    <transport>TCP5060 UDP5060 TCP80 TCP443</transport>
    <!-------- the list of ports the proxy is listening to-->
  </location>
  <location>
    <node_IP>192.168.0.100</node_IP>
    <!-------- the IP address at which the user can be reached-->
    <transport>TCP5060 UDP5060 TCP80 TCP443</transport>
    <!-------- the list of ports the proxy is listening to-->
  </location>
</resource>
```

The resource name for a proxy location record is "proxy location"+

9. P2P Overlay API

The P2P overlay layer provides an API by which requests for resource placement and search over the P2P overlay network can be made. The semantics of the core functions of the API are described below:

```python
get(in overlay_id, in key, out records, out error)
  - action: look up a resource from the P2P overlay network identified by the overlay_id.
```
- input parameter
  overlay_id: An identifier that uniquely identifies a particular overlay the node is a member.
  key: the key of the target resource.
- output parameters
  records: records of the resources returned from the search.
  One or multiple records may be included.
  error: an error code.

add(in overlay_id, in key, in record, in lifetime, in option, out error)
- action: place a resource in the P2P overlay network. If a resource of the same key exists in the overlay network already, the resource given in the call is added in the network in addition to the existing resource. That is, add() cannot be used to change existing resources in the overlay network.
- input parameters
  overlay_id: An identifier that uniquely identifies a particular overlay the node is a member.
  key: the key of the resource to be placed.
  record: the record of the resource to be placed.
  lifetime: lifetime of the resource to be placed.
  option: option for placement.
  NO_UPDATE_OPTION: Indicates that the record is not updatable; then no credentials to identify the source need to be stored along with record. If updatable, a security credential (likely derived from the security credentials generated during authentication procedure) is associated with the record.
- output parameters
  error: the error code.

update(in overlay_id, in key, in record, in lifetime, in option, out error)
- action: update a resource in the P2P overlay network and create one with the given record if a resource with the given key does not exist yet. The update operation is allowed for existing resources whose owner (or creator) is the same as the updating user. This is verified based on security credentials. If the stored resource doesn’t contain any associated security credentials the update operation will fail.
- input parameters
  overlay_id: An identifier that uniquely identifies a particular overlay the node is a member.
  key: the key of the resource to be placed.
  record: the record of the resource to be placed.
  lifetime: lifetime of the resource to be placed.
option: option for placement.
- output parameters
  error: the error code.

remove(in overlay_id, in key, out error)
- action: remove the resources of the given key from the overlay network. Like the update function, this is allowed for existing resources whose owner (or creator) of the resources is the same as the requesting user. If the authentication doesn’t succeed or if there is no associated security credentials, the remove operation will fail.
- input parameter
  overlay_id: An identifier that uniquely identifies a particular overlay the node is a member.
  key: the key of the target resource.
- output parameters
  error: an error code.

If anyone is allowed to remove or update location records of other users, it becomes so easy to block or interfere with incoming calls to a particular user. So it is imperative to give the owner of a resource record the authority to set access control policies for the resource record, in particular, about updating and removing the resource record. To support the above API, the P2P overlay layer manages resource records based on the resource key and resource’s owner identity. Conceptually the overlay layer stores every resource record with a control information record that includes the resource key, the owner identity, and the access control policy set by the owner.

10. What Needs To Be Specified

For interoperation between different devices, the following interfaces MUST be specified: the SN-ON interface, the SN-SN interface, the ON-Login Server interface, the SN-login server interface, and the peer-bootstrap server interface. Since nodes startup as ON and complete bootstrap before changing to a SN, there is only one interface with the bootstrap server.

And the formats of resource records for SIP MUST be specified. Section 8 describes two example resource record types and their formats.

The API (application program interface) of the P2P overlay layer, like that in Section 9, is not required to be specified in the syntactical level that depend on implementations of the SIP layer and the P2P overlay on each device. However, there MUST be a common set of services the SIP layer can receive from the P2P overlay layer,
which can be captured in a semantic definition of the P2P overlay API like that in Section 9.

11. Security Considerations

Peer-to-peer networks need to balance between providing appropriate trust, and reducing requirements for centralized authority. Particularly, it is important not to aim at solving generalized distributed trust problems but instead to focus on use-cases appropriate to endpoint identification and session establishment.

11.1 Bootstrapping Security

Becoming an ordinary node requires use of dynamically configured super node information. Where SN information is received from an untrusted source, or over an unsecured channel, care should be taken in attempting to contact these nodes, so that time and resources are not wasted contacting bogus servers.

Additionally, care should be taken not to unduly extend trust to discovered devices, even where the origin of the SN information is trusted. This is important, as there may be a change in the trust of a particular node between the time which SN information was gathered and the time it is used by the ON for bootstrapping. As such, it is important that nodes authenticate peers in order to establish relationships.

11.2 ON/SN Authentication

When connecting to the network, a node needs to authenticate that its selected SN is trusted, and that it is talking directly to that node. Similarly, the SN needs to identify that the ON is a valid participant in the network, and that it is authorized to perform lookup, and record update operations for P2P SIP.

This authentication needs to occur without constant reference to a single login server, which would limit scalability, introduce a single point of failure and therefore defeat the peer-to-peer paradigm.

11.3 Peer Transport Security

Message authentication must be used between one node and another in a peer-to-peer network. Where privacy against non-P2P participants is required, encryption should also be used. Use of encryption suites with key negotiation may achieve not only the requirement for mutual authentication, but also data transport security as well [6][7][8].
11.4 Firewalls

Firewalls are typically configured by an organization in order to prevent certain classes of traffic passing across administrative boundaries. While application-level gateways exist which control message contents at upper layers, most firewalls control data flows by allowing data transfer on ports assigned to specific services. Any firewall traversal technique that uses an assigned port purely to find an open channel through a gateway is therefore likely to be contravening the security policy of the peer’s network. This may particularly be the case where external devices such as tunnel servers allow inbound session initiation through port relay [10].

Additionally, operation of SN services on ports normally preserved for other purposes may expose that node to access or attack from outside the intended boundary of the peer-to-peer network, as such ports may not be blocked by a firewall.

11.5 Network Address Translators

Peer-to-peer session establishment through a NAT or firewall may require tunneling assistance from a relay device on the public Internet. If only a small number of relay devices are available, they are a potential single-point of failure, and may be subject to denial of service attacks. Relay devices pose a particular threat in that compromise of a peer’s relay may allow packet insertion or modification. As such, message exchanges SHOULD authenticate the peer’s credentials, rather than relying entirely on channel security.

11.6 Registration

When a SIP entity places a location record in the P2P overlay network to register a URI, the information in the record controls the address at which it can be contacted. Modifications to such information need to be authenticated, in order to ensure communications sessions are not redirected, or subject to man-in-the-middle attacks. This requires that SIP entities generating location records SHOULD sign their own location record updates, and that updates with different credentials do not overwrite each other.

Additionally, these signed update messages need to provide a freshness information, in the form of a monotonically increasing number (which may be the origin peer’s clock), to ensure that any update operations do not replace newer registrations.

Registration of a device’s address may also indicate its location in the physical world. Where this is an important consideration, adoption of a derived transport address from a TURN server may allow...
a level of indirection and privacy [10].

11.7 Session Endpoint Discovery and Security

Where no central servers are required to identify endpoints, centralized mechanisms for identifying valid SIP transport and security are no longer applicable [9]. Instead, peers SHOULD use the credential information associated with the other peer to verify the address, port, and preference information for particular SIP URIs advertised in the DHT.

The content or origin of get() commands may divulge information about the location or identity of the requester. In order to maintain privacy, a SN may remove identifying information when making a request on another’s behalf. Rate limitation of the requests from a source may be performed in order to limit abuse and overload of the P2P network.

11.8 Ill Behavior of Ordinary Nodes

Ordinary nodes do not have direct access to the DHT, but they are able to read and place data into it. As such, they gain many of the advantages of being on the overlay, without contributing to its maintenance. As maintaining the DHT requires some bandwidth, computing and storage resources, some ONs may wish to not become SNs, even if this would benefit the overlay network’s operation. While nodes are to be encouraged to transform into SNs, it is in practice hard to enforce them to take on the SN role.

Any node may place useless or distracting information on the overlay, through generation of update() and add() commands. Unless user credentials are used as the basis of checks to update or add new information, it may be possible to remove other users’ legitimate data, which may have further security consequences. If two devices purport to add information pertaining to the same record, both need to be retained, distinguished by their credentials, as the SN performing DHT operations will typically be unable to identify which is correct.

As mentioned before, updates of the hash table are likely to be more computationally onerous for SNs, as authenticity checks may be required in order to modify records. As such, it may be possible to deny service on a portion of the hash table by continually sending the SN update() and add() operations.

In some DHT systems, knowledge of routing mechanisms may be used to generate get() queries which increase the processing delays across the entire overlay network, or focus additional traffic at a
particular SN with constrained resources.

11.9 Ill Behavior of Super Nodes

A super node is a device with direct ability to change the topology of the overlay network. It is able to receive read-only and modifying operations on the network, and may be able to insert, modify, drop, or selectively transmit the messages it receives. It is responsible for maintaining the data sent to it, and passing a portion of the hash table to its peers upon joining or leaving the network. As such, the SN has a great amount of influence and power within a P2P network, and compromise of an SN (or a collection of SNs) seriously jeopardizes the operation of the network.

Therefore, it is very important to use strong authentication and authorization to ensure that participant SNs are valid candidates to join the DHT. Additionally, it is important to maintain additional backups for data that may be under control of an SN that goes bad. Use of origin authentication, integrity and freshness checks may at least ensure that items stored in the DHT are unmodified upon storage.

12. Acknowledgements

The authors would like to thank Salman Basset and Henning Schulzrinne for the discussion with them on P2P SIP architecture issues and their comments. Their work provided the authors with valuable information on a large scale P2P Internet telephony system.

13. References

13.1 Normative References


13.2 Informative References


Authors’ Addresses

Eunsoo Shim
Panasonic Digital Networking Laboratory
Two Research Way, 3rd Floor
Princeton, NJ  08540
USA

Email: eunsoo@research.panasonic.com
Sathya Narayanan  
Panasonic Digital Networking Laboratory  
Two Research Way, 3rd Floor  
Princeton, NJ  08540  
USA  

Email: sathya@research.panasonic.com

Greg Daley  
Panasonic Digital Networking Laboratory  
Two Research Way, 3rd Floor  
Princeton, NJ  08540  
USA  

Email: gregd@research.panasonic.com
Intellectual Property Statement

The IETF takes no position regarding the validity or scope of any Intellectual Property Rights or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; nor does it represent that it has made any independent effort to identify any such rights. Information on the procedures with respect to rights in RFC documents can be found in BCP 78 and BCP 79.

Copies of IPR disclosures made to the IETF Secretariat and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this specification can be obtained from the IETF on-line IPR repository at http://www.ietf.org/ipr.

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights that may cover technology that may be required to implement this standard. Please address the information to the IETF at ietf-ipr@ietf.org.

Disclaimer of Validity

This document and the information contained herein are provided on an "AS IS" basis and THE CONTRIBUTOR, THE ORGANIZATION HE/SHE REPRESENTS OR IS SPONSORED BY (IF ANY), THE INTERNET SOCIETY AND THE INTERNET ENGINEERING TASK FORCE DISCLAIM ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Copyright Statement

Copyright (C) The Internet Society (2006). This document is subject to the rights, licenses and restrictions contained in BCP 78, and except as set forth therein, the authors retain all their rights.

Acknowledgment

Funding for the RFC Editor function is currently provided by the Internet Society.