Internet Engineering Task Force   O. Akonjang
Internet-Draft   A. Feldmann
Intended status: Informational   DT Labs/TU Berlin
Expires: September 3, 2009   S. Previdi
B. Davie
Cisco Systems, Inc.
D. Saucez
Universite catholique de Louvain
March 2, 2009

The PROXIDOR Service
draft-akonjang-alto-proxidor-00.txt

Status of This Memo

Copyright Notice

Abstract

Several applications, such as peer-to-peer (P2P), content distribution and realtime services rely on selection mechanisms in order to select the peer or server from which to request the service. Examples of such services are: file sharing, media streaming and voice gateways.

Application-layer selection algorithms do not typically take into account network-layer topology information; either that information is unavailable to them, or when such information is available (e.g., from BGP Looking Glass servers), it does not include sufficient information about the local topology in the neighbourhood of the application client(s). Therefore, most applications today make their selection decisions based on performance measurements (combined with some amount of random selection) and largely ignore network layer routing. It has been demonstrated that by keeping the traffic local (e.g., within the same Autonomous System) both infrastructure utilization and application performance may be improved.

By enhancing selection algorithms through the use of accurate network-layer topology, applications may improve performance while network operators are also able to reduce the utilization of infrastructure resources by application traffic. At the same time, exchange of information between the application and the network should not be allowed to compromise confidentiality for either party. Detailed routing information owned by the service provider should not be made publicly available, while detailed information about the application should also not be made known to the service provider.

This draft introduces a signaling protocol which we call "PROXIDOR". The PROXIDOR protocol is a request-response protocol in which a PROXIDOR Client (PxC) issues requests to and receives responses from a PROXIDOR Server (PxS). The questions of how a PxC discovers a PxS and how a PxS acquires network-layer topology information are beyond the scope of this document.

Table of Contents

1. Introduction

This draft introduces the concept of a PROXIDOR service (PS). A PROXIDOR Service (PS) aims to deliver to selection algorithms at the application layer (or any other consumer of the PROXIDOR service) the topological guidance that will allow an improved selection scheme, taking into considerations the topology and infrastructure that is going to be used for transmitting data from/to a given selected peer, host or server. The protocol defined in this draft aims to be generic and exploitable by any application entity requiring such a service, no matter its architecture and implementation.

This draft describes a signaling protocol through which a PROXIDOR Client (PxC) requests service from a PROXIDOR Server (PxS). In the current version of the document, the protocol is described only in high-level, abstract terms.

As an example, a peer-to-peer client, once it has obtained from the application layer the list of peers through which a given content or service can be obtained, requests PROXIDOR services from a PROXIDOR server. The PROXIDOR query is built with the list of candidate peers' IP addresses and sent to the PxS. The PxS replies with the original list of IP addresses sorted by preference.

The definition of "preference" in this context requires some further explanation. By assigning a higher preference to a particular address, the PxS indicates to the application that, all things being equal, the application should prefer that address to other addresses of lower preference. Exactly what the PxS means by higher preference, or how this preference is calculated, is intentionally not made explicit. As an example, preference may be calculated by using routing metric distance, with nodes that are a shorter distance from the requester being given a higher preference than those that are at a greater distance. Such a calculation could, for example, be performed by inspecting network-layer information (IGP, BGP, etc) and may also be influenced by policies of the service provider.

We assume that the service provider cannot force the application to adhere to the preferences that the PROXIDOR service provides. Therefore, the service provider has an incentive to provide information that is useful to the application; otherwise it is likely to be ignored.

Although we use IP addresses in the preceding example, similar ranking operations could also be performed on AS numbers, address prefixes, etc. While it is necessary to ensure interoperability through the standardization of the signaling protocol, there is no immediate need to standardize any algorithm or any computational

method that computes ranks or preferences.

1.1. Historical Background of this Proposal

The architecture and protocol described in this document arose from the merger of three previous pieces of work: the Oracle service [1], the ISP-Driven Informed Path Selection (IDIPS) service [5] and the Proximity service [6]. The name PROXIDOR, like the proposal itself, contains elements of each of these three original work items.

2. Conventions

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

3. Benefits of the PROXIDOR Service

The PROXIDOR service offers many benefits to both end-users (consumers of the service) and Internet Service Providers (ISPs) (providers of the service).

3.1. Benefits to End-users

Hosts that use the PROXIDOR service can benefit in multiple ways, e.g., they no longer need to infer topology information or measure path performance by themselves. They can take advantage of the knowledge of the ISP to avoid bottlenecks and boost performance.

3.2. Benefits to the ISP

ISPs also benefit from the PROXIDOR service. It enables them to influence the neighborhood selection process of overlay networks to e.g. ensure locality of traffic and also regain the ability to efficiently engineer traffic that traverses backbone and transit links, thus allowing a better provisioning of the networking infrastructure.

4. The PROXIDOR Architecture

The PROXIDOR architecture can be defined in terms of its entities, the set of messages that are exchanged between these entities and the rules governing the exchanges.

4.1. Definitions and Terminologies

Autonomous System (AS):
A single network or a collection of networks under the same administrative control.

Autonomous System Number (ASN):
A globally unique number, used to identify an AS and to exchange exterior routing information with other (neighboring) ASes.

PROXIDOR client (PxC):
An Internet end-host that requests for PROXIDOR services by creating and sending queries.

PROXIDOR server (PxS):
An Internet system that accepts and processes PROXIDOR queries.

Metric:
Criteria used by server to process/rank IP addresses, prefixes or ASNs in a query.

PROXIDOR Query (PxQ):
A message to request for a particular PROXIDOR service.

PROXIDOR Response (PxR):
An answer to a PROXIDOR query.

PROXIDOR Error (PxE):
Indicates non-conformity.

PROXIDOR Failure (PxF):
A PROXIDOR system malfunction.

PROXIDOR Services (PS):
A PROXIDOR service that can be directly requested by clients.

PROXIDOR Service Protocol (PSP):
The protocol used by PROXIDOR entities to communicate with each oher.

PROXIDOR Source List (PSL):
A list of sources that is sent to PxS for ranking.

PROXIDOR Target List (PTL):
A list of targets that is sent to PxS for ranking.

PROXIDOR Couples (PC):
A ranked list of <source, destination>.

PROXIDOR Source Address (PSA):
Source IP or Prefix address used in PxQ when requesting for PROXIDOR services.

Type Length Value (TLV):
Data encoding format for PROXIDOR payloads.

4.2. Scope

This draft defines the PROXIDOR service, describes its functionality and gives details of its architecture. Details of the PROXIDOR protocol itself (i.e., details of the messages that are exchanged between entities of the PROXIDOR service) are out of scope of this document.

4.3. Design Goals

We expect the PROXIDOR service to be requested by a high number of simultaneous clients and therefore aspects such as simplicity, scalability, performance and flexibility are among the most important criteria:

o Simplicity. The protocol must be lightweight in its content and state machinery. Different flavors may be defined in order to cope with different application scenarios. However, in most of practical cases, the protocol should carry the minimal set of information in its simplest form of encoding.

o Scalability. The protocol is expected to be used by a large amount of consumers simultaneously, which means that providers will have to absorb a large amount of transactions. Protocol specification should take into account the scalability factors that would allow the protocol and its implementations to scale.

o Performance. Related to scalability, PROXIDOR servers will have to handle a high number of simultaneous transactions and therefore the performance of handling protocol packets is critical for the overall scalability and deployability of the protocol.

o Flexibility. The protocol should allow different modes of operations with different encoding techniques.

4.4. Design Choices

Two major concerns of the PROXIDOR protocol design are scalability and performance. A server (or a cluster of servers) should be able to simultaneously handle a very large number of client requests and yet, suffer no penalties in terms of performance. To address these

concerns, important design choices need to be made, including;

i) Choice of transport protocol:

The PROXIDOR service SHOULD be able to function with UDP, TCP and HTTP.

ii) Choice of approach:

Although most aspects of the PROXIDOR protocol are new, advantage is still taken of already existing protocols, such as the Domain Name System (DNS) protocol. In principle, we see a lot of functional similarities between the DNS and the PROXIDOR service. The DNS is not only distributed and very scalable, but can also effectively handle many queries from a large number of clients simultaneously. The DNS protocol approach is thus partly adopted.

Further concerns of the protocol design are flexibility and extensibility. The protocol should be flexible enough to adapt to changes in end-users' and ISPs' needs and be extensible enough to incorporate new types of services without the need to undergo a fundamental change.

To address these concerns, PROXIDOR uses both Service Codes (to request for particular PROXIDOR services or to express their preferences) and Service Categories (to create and categorize (new) PROXIDOR service groups).

5. Entities of the PROXIDOR System

A PROXIDOR system consists of PROXIDOR clients and PROXIDOR servers. Both entities implement the PROXIDOR protocol in order to interact with each other. Each entity independently plays an important role in the overall system architecture.

5.1. The PROXIDOR Client

The PROXIDOR client can either be an end-user host that generates and sends queries to a PROXIDOR server or a PROXIDOR server that generates and/or forwards queries to other PROXIDOR servers.

To use the PROXIDOR service, a client (PxC) must generate a standard PROXIDOR query (PxQ) and send to the PROXIDOR server (PxS). It can also optionally cache a copy of the sent packet. It MUST clear its cache before attempting to contact another PROXIDOR server.

5.2. The PROXIDOR Server

A PROXIDOR server that receives a query, can generally react in one of the following ways:

- grant the request by responding appropriately to the query,
- silently drop the query if it is e.g., running out of resources,
- send back a PROXIDOR error message (PxE), in case of a malformed request,
- send back a PROXIDOR failure message (PxF), in case of system malfunction,
- send back a re-direct message, re-directing the client to another PROXIDOR server, if it doesn't offer the requested service and knows another (trusted) server that does.

The server MAY also decide to explicitly express the criteria it used in ranking the contents of the response message. It should be noted that revealing the criteria does not directly reveal the algorithm or functions that these criteria were used to construct.

The PROXIDOR protocol may also be used between servers. When a server needs more accurate information that is not available in its database, it may also use the PROXIDOR service to request this information from another server. This could be typical in environments such as the global coordinate system in [2]. PROXIDOR protocol supports authentication between a client and a server and between servers. Authentication could be used to establish a kind of trust between PROXIDOR entities involved in a transaction.

5.3. The Ranking System

The most common (perhaps the default) criteria for the ranking system is the minimal distance between the requester and each individual IP address in the query message. In this case the PxS evaluates the distance between requester and each address of the list and returns the list in an ordered fashion. Distance is evaluated according to the network-layer topology.

A different preference criteria can be specified, relying on AS membership: Given an unsorted list of IP addresses, the PxS returns the list ordered by preference in AS membership. The ranked list starts with addresses belonging to the same AS of the requester, then all other addresses ordered according to the number of AS-hops between their AS and requester's AS.

The ranking system can also use criteria that are based on measured performance, such as available bandwidth and link delays, as well as those that are based on the ISP's private policies and need to be taken into consideration.

Criteria MAY be used either alone or in combinations, when evaluating preferences.

While a ranking is one way to assign preferences to addresses, prefixes, etc., it may also be desirable to assign a weight to the elements in the list, to indicate more clearly the extent to which some elements are preferred over others. This approach is described in [4].

The semantic of each ranking request is carried within the query message (implicitly or explicitly). However, the server MAY or MAY NOT take this into account.

6. PROXIDOR Messages

There are generally two types of PROXIDOR messages; the PROXIDOR query (PxQ) message and the PROXIDOR response (PxR) message.

6.1. The Query Message

A standard (default) PROXIDOR query (PxQ) is that which is sent from a PROXIDOR client (PxC) to a PROXIDOR server (PxS). A PROXIDOR server (PxS) can also query another PxS using the same message format. The PROXIDOR protocol differentiates between these two query types.

Query messages may also carry requests that express desires for more specific information or services. The server is not compelled to grant them. The server MAY decide to grant or ignore them, depending on its own preferences or individual policies. For example, it can decide to grant server-generated desires sent by trusted PxS peers and ignore client-generated ones that it considers to be too intrusive.

6.2. The Response Message

Response messages are similar in structure to query messages. Response messages can make use of attributes (optional) to send additional information about designated payload types, e.g., when responding to queries sent by a trusted PxS peer. Such peers could exist within an AS or in a global coordinate system, made up of PROXIDOR servers that are managed by different ISPs. The additional information supplied by the use of attributes can help the receiving

server make better decisions before responding to client-generated queries.

The order of the items in the response message is very important. The first item indicates highest priority or preference and the last one, least priority or preference. There is no general derivable relationship between this order and the criteria used to create it, since the latter MAY depend on factors that are of the PROXIDOR Service Provider's individual choice.

Although some characteristics of the response message is also determined by the same factors that affect those of the query message, all response packets MUST additionally adhere to the important factor of strict ranking. Ranking could force the response payload to be packaged in particular orders relative to each other, requiring the use of extra bytes in the response message than that used in the query. The details of this aspect is out of scope of this document.

7. Use Cases

Use cases are grouped according to service categories, such as the Oracle service category, the Proximity service category and the IDIPS service category.

7.1. The Oracle Use Case

i) The PROXIDOR service is used in a P2P environment to influence the neighborhood selection process.

An end-host that wants to join a P2P overlay, first needs to locate and cretae a list of potential neighbors. Instead of randomly connecting to clients in the list, the end-host can use the PROXIDOR service to establish a more effective connection pattern.

1. The PROXIDOR client (PxC) creates a PROXIDOR Target List (PTL) from the list of potentials neighbors.

2. PXC contructs a PROXIDOR query message (PxQ) with the PTL as payload.

3. PxC then sends PxQ to a PROXIDOR server (PxS) for ranking.

4. PxS receives and extracts PTL from PxQ. It uses its knowledge of the network to re-arrange the list, ranking them according to some pre-defined criteria, e.g. locality and bandwidth.

   + The list is ranked according to priority (or preference), i.e., from highest to lowest.

   + Proximal and optimal performing neighbors (relative to PxC) are given highest priority and are placed at the top of the list.

   + PxS can decide to supplement the response with extra information through the use of attributes and their values.

5. PxS then sends the ranked PTL back to PxC.

6. PxC can now use the ranked PTL to establish optimal overlay connections.

ii) The PROXIDOR service is used in a P2P environment to influence the selection of sources from where content is downloaded.

After a successful search or after joining a swarm, PxC may have multiple sources from which content could be downloaded. It can use the PROXIDOR service again to locate the best sources (from among these potential sources) to download from.

1. PxC creates a new PTL from the list of potential sources.

2. PxC constructs a new PxQ using the new PTL.

3. PxC sends the new PxQ to PxS.

4. PxS extracts and ranks the PTL according to optimal performance.

5. PxS sends the ranked PTL back to PxC.

6. PxC can now use the ranked PTL to download from the best ranked sources in a more efficient mannner.

7.2. The Proximity Use Case

1. PxC obtains from the application layer the list of servers/peers where data or service is available. PxC builds a list of IP addresses corresponding to these peers/servers.

2. PxC builds a PROXIDOR query (PxQ). PxC insert following information in the query:

   * PSA: the PxC IP address.

   * PTL: the list of target IP addresses.

   * PRC: set to IP-Address-based or AS-based.

3. PxC sends the PxQ to the PxS.

4. PxS receives PxQ and extracts the following information:

   * PROXIDOR Source Address (PSA). PSA is the source IP address of the requester.

   * PROXIDOR Target List (PTL). PTL is the list of IP addresses for which PROXIDOR service is requested.

   * PROXIDOR Ranking Criteria.

5. PxS computes PROXIDOR algorithms and ranks the PTL based on PRC. Preference is determined according to network-layer topology information.

6. PxS creates a PxR message including:

   * PROXIDOR Source Address (PSA): the PSA address as received in the request packet.

   * PROXIDOR Target List (PTL): the ordered (ranked) original PTL (as received in the request packet).

   * PROXIDOR Ranking Criteria: same as PRC in PxR message.

7. PxC receives PxR, extracts PTL and may apply it to its selection scheme.

7.3. The IDIPS Use Case

The PxC is connected to its network with, at the same time, a wireless and a wired connection.

1. PxC obtains from the application layer the list of servers/peers where data or service is available. PxC builds a list of IP addresses corresponding to these peers/servers.

2. PxC builds a PROXIDOR query (PxQ). PxC insert following information in the query:

   * PSL: the list of PxC IP addresses.

   * PTL: the list of target IP addresses.

   * PRC: set to delay-based.

3. PxC sends the PxQ to the PxS.

4. PxS receives PxQ and extracts the following information:

   * PROXIDOR Source List (PSL). PSL is the list of IP addresses of the requester.

   * PROXIDOR Target List (PTL). PTL is the list of IP addresses for which PROXIDOR service is requested.

   * PROXIDOR Ranking Criteria.

5. PxS determines the feasible <source,destination> pairs where souces are in PSL and destinations are in PTL. PxS computes PROXIDOR algorithms and ranks the pairs. Preference is determined according to network-layer topology information.

6. PxS creates a PxR message including:

   * PROXIDOR Couples (PC): the ordered (ranked) list of <source, destination> pairs built with the original PSL and the original PTL.

   * PROXIDOR Ranking Criteria: same as PRC in PxQ message.

7. PxC receives PxQ, extracts PC and may apply it to its selection scheme.

8. Extensibility

The protocol is capable of using different transport mechanisms and also allows both clients and servers to express particular desires. These aspects add a great deal of flexibility to the protocol construct.

The use of service categories also creates enough room for extensibility when the need for such arises, e.g., when changes in end-users' or ISPs' needs necessitate the creation of additional service features or even totally new service groups.

9. Security Considerations

The PROXIDOR system MUST ensure that data is exchanged between PROXIDOR servers in a secure manner. Details of this aspect and

further security considerations will be treated in future versions of this document.

10. Contributors

We acknowledge with much thanks the enormous contributions made through discussions, remarks and suggestions by the following individuals (in alphabetical order): Vinay Aggarwal, Olivier Bonaventure, Pierre Francois, Benjamin Frank, Luigi Iannone, Jun Jiang, Ingmar Poese, Georgios Smaragdakis and Pengchun Xie.

11. References

11.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

11.2. Informative References

[1] Aggarwal, V., Feldmann, A., and C. Scheideler, "Can ISPs and P2P systems co-operate for improved performance?", ACM SIGCOMM Computer Communications Review (CCR), 37(3):29-40, July 2007.

[2] Aggarwal, V., Feldmann, A., and R. Karrer, "An Internet Coordinate system to enable collaboration between ISPs and P2P systems", In Proceedings of the 11th International ICIN Conference , October 2007.

[3] Aggarwal, V., Akonjang, O., and A. Feldmann, "Improving User and ISP Experience through ISP-aided P2P Locality", In Proceedings of 11th IEEE Global Internet Symposium 2008 (GI '08) , 2008.

[4] Shalunov, S., Penno, R., and R. Woundy, "ALTO Information Export Service", draft-shalunov-alto-infoexport-00 (work in progress) , October 2008.

[5] Saucez, D., Donnet, B., and O. Bonaventure, "IDIPS: ISP-Driven Informed Path Selection", draft-saucez-alto-idips-01 (work in progress) , February 2008.

[6] Previdi, S., "Routing Proximity Services", IETF 73, http://www.ietf.org/proceedings/08nov/slides/alto-0.pdf, November 2008.

Authors' Addresses

Obi Akonjang
DT Labs/TU Berlin
Ernst-Reuter-Platz 7
10587 Berlin
Germany

EMail: obi@net.t-labs.tu-berlin.de

Anja Feldmann
DT Labs/TU Berlin
Ernst-Reuter-Platz 7
10587 Berlin
Germany

EMail: anja@net.t-labs.tu-berlin.de

Stefano Previdi
Cisco Systems, Inc.
Via Del Serafico
Rome 00142
Italy

EMail: sprevidi@cisco.com

Bruce Davie
Cisco Systems, Inc.
1414 Mass. Ave.
Boxborough, MA 01719
USA

EMail: bsd@cisco.com

Damien Saucez
Universite catholique de Louvain
Place Sainte Barbe 2
Louvain-la-Neuve, 1348
Belgium

EMail: damien.saucez@uclouvain.be