Network Working Group
Request for Comments: 2843
Category: Informational
P. Droz
IBM
T. Przygienda
Siara
May 2000

Proxy-PAR

Status of this Memo

This memo provides information for the Internet community. It does not specify an Internet standard of any kind. Distribution of this memo is unlimited.

Copyright Notice

Abstract

Proxy-PAR is a minimal version of PAR (PNNI Augmented Routing) that gives ATM-attached devices the ability to interact with PNNI devices without the necessity to fully support PAR. Proxy-PAR is designed as a client/server interaction, of which the client side is much simpler than the server side to allow fast implementation and deployment.

The purpose of Proxy-PAR is to allow non-ATM devices to use the flooding mechanisms provided by PNNI for registration and automatic discovery of services offered by ATM attached devices. The first version of PAR primarily addresses protocols available in IPv4. But it also contains a generic interface to access the flooding of PNNI. In addition, Proxy-PAR-capable servers provide filtering based on VPN IDs [1], IP protocols and address prefixes. This enables, for instance, routers in a certain VPN running OSPF to find OSPF neighbors on the same subnet. The protocol is built using a registration/query approach where devices can register their services and query for services and protocols registered by other clients.

1 Introduction

In June of 1996, the ATM Forum accepted the "Proxy-PAR contribution as minimal subset of PAR" as a work item of the Routing and Addressing (RA) working group, which was previously called the PNNI working group [2]. The PAR [3] specification provides a detailed description of the protocol including state machines and packet formats.

The intention of this document is to provide general information about Proxy-PAR. For the detailed protocol description we refer the reader to [3].

Proxy-PAR is a protocol that allows various ATM-attached devices (ATM and non-ATM devices) to interact with PAR-capable switches to exchange information about non-ATM services without executing PAR themselves. The client side is much simpler in terms of implementation complexity and memory requirements than a complete PAR instance. This should allow an easy implementation on existing IP devices such as IP routers. Additionally, clients can use Proxy-PAR to register various non-ATM services and the protocols they support. The protocol has deliberately been omitted from ILMI [4] because of the complexity of PAR information passed in the protocol and the fact that it is intended for the integration of non-ATM protocols and services only. A device executing Proxy-PAR does not necessarily need to execute ILMI or UNI signalling, although this will normally be the case.

The protocol does not specify how a client should make use of the obtained information to establish connectivity. For example, OSPF routers finding themselves through Proxy-PAR could establish a full mesh of P2P VCs by means of RFC2225 [5], or use RFC1793 [6] to interact with each other. LANE [7] or MARS [8] could be used for the same purpose. It is expected that the guidelines defining how a certain protocol can make use of Proxy-PAR should be produced by the appropriate working group or standardization body responsible for the particular protocol. An additional RFC [9] describing how to run OSPF together with Proxy-PAR is published together with this document.

The protocol has the ability to provide ATM address resolution for IP-attached devices, but such resolutions can also be achieved by other protocols under specification in the IETF, e.g. [10]. Again, the main purpose of the protocol is to allow the automatic detection of devices over an ATM cloud in a distributed fashion, omitting the usual pitfalls of server-based solutions. Last but not least, it should be mentioned here as well that the protocol complements and coexists with the work done in the IETF on server detection via ILMI extensions [11,12,13].

2 Proxy-PAR Operation and Interaction with PNNI

The protocol is asymmetric and consists of a discovery and query/registration part. The discovery is very similar to the existing PNNI Hello protocol and is used to initiate and maintain communication between adjacent clients and servers. The registration and update part execute after a Proxy-PAR adjacency has been established. The client can register its own services by sending registration messages to the server. The client obtains information it is interested in by sending query messages to the server. When the client needs to change its set of registered protocols, it has to re-register with the server. The client can withdraw all registered services by registering a null set of services. It is important to note that the server side does not push new information to the client, neither does the server keep any state describing which information the client received. It is the responsibility of the client to update and refresh its information and to discover new clients or update its stored information about other clients by issuing queries and registrations at appropriate time intervals. This simplifies the protocol, but assumes that the client will not store and request large amounts of data. The main responsibility of the server is to flood the registered information through the PNNI cloud such that potential clients can discover each other. The Proxy-PAR server side also provides filtering functions to support VPNs and IP subnetting. It is assumed that services advertised by Proxy-PAR will be advertised by a relatively small number of clients and be fairly stable, so that polling and refreshing intervals can be relatively long.

The Proxy-PAR extensions rely on appropriate flooding of information by the PNNI protocol. When the client side registers or re-registers a new service through Proxy-PAR, it associates an abstract membership scope with the service. The server side maps this membership scope into a PNNI routing level that restricts the flooding. This allows changes of the PNNI routing level without reconfiguration of the client. In addition, the server can set up the mapping table such that a client can flood information only to a certain level. Nodes within the PNNI network take into account the associated scope of the information when it is flooded. It is thus possible to exploit the PNNI routing hierarchy by announcing different protocols on different levels of the hierarchy, e.g. OSPF could be run inside certain peer groups, whereas BGP could be run between the set of peer -groups running OSPF. Such an alignment or mapping of non-ATM protocols to the PNNI hierarchy can drastically enhance the scalability and flexibility of Proxy-PAR service. Figure 1 helps visualize such a scenario. For this topology the following registrations are issued:

```
    +-+
    | | PNNI peer group    # PPAR capable  @ PNNI capable  * Router
    +-+                      switch          switch


                   Level 40
                   +---------------------------+
                   |                           |
                   |                           |
                   |      @ ---- @ ---- @      |
                   |      |             |      |
                   +----- | ----------- | -----+
                          |             |
           Level 60       |             |
           +------------- | ---+    +-- | --------------+
           |              |    |    |   |               |
      R1* ------#-P1------@    |    |   @---------P3-#------- * R3
           |              |    |    |   |               |
      R2* ------#-P2------+    |    |   +---------P4-#------- * R4
           |                   |    |                   |
           +-------------------+    +-------------------+
```

Figure 1: OSPF and BGP scalability with Proxy-PAR autodetection (ATM topology).

1. R1 registers OSPF protocol as running on the IP interface 1.1.1.1 and subnet 1.1.1/24 with scope 60

2. R2 registers OSPF protocol as running on the IP interface 1.1.1.2 and subnet 1.1.1/24 with scope 60

3. R3 registers OSPF protocol as running on the IP interface 1.1.2.1 and subnet 1.1.2/24 with scope 60

4. R4 registers OSPF protocol as running on the IP interface 1.1.2.2 and subnet 1.1.2/24 with scope 60

and

1. R1 registers BGP4 protocol as running on the IP interface 1.1.3.1 and subnet 1.1/16 with scope 40 within AS101

2. R3 registers BGP4 protocol as running on the IP interface 1.1.3.2 and subnet 1.1/16 with scope 40 within AS100

For simplicity the real PNNI routing level have been specified, which are 60 and 40. Instead of these two values the clients would use an abstract membership scope "local" and "local+1". In addition, all registered information would be part of the same VPN ID.

Table 1 describes the resulting distribution and visibility of registrations and whether the routers not only see but also utilize the received information. After convergence of protocols and the building of necessary adjacencies and sessions, the overlying IP topology is illustrated in Figure 2.

```
                     AS101         DMZ      AS100
                   #########                ##########
                           #                #
               |           #   |            #            |
               +-- R1 ---------+            #       R4 --+
               |           #   |            #            |
               |           #   | BGP4 on    #    OSPF on |
               | OSPF on   #   | subnet     #     subnet |
               | subnet    #   | 1.1/16     #   1.1.2/24 |
               | 1.1.1/24  #   |            #            |
               |           #   +------------------- R3 --+
               +-- R2      #   |            #            |
               |           #                #
                   #########                ##########
```

Figure 2: OSPF and BGP scalability with Proxy-PAR autodetection (IP topology).

Expressing the above statements differently, one can say that if the scope of the Proxy-PAR information indicates that a distribution beyond the boundaries of the peer group is necessary, the leader of a peer group collects such information and propagates it into a higher layer of the PNNI hierarchy. As no assumptions except scope values can normally be made about the information distributed (e.g. IP addresses bound to AESAs are not assumed to be aligned with them in any respect), such information cannot be summarized. This makes a careful handling of scopes necessary to preserve the scalability of the approach as described above.

Reg# Router#	1.	2.	3.	4.	5.	6.
R1	R	U			R	U
R2	U	R			Q	Q
R3			R	U	R	U
R4			U	R	Q	Q

R registered
Q seen through query
U used (implies Q)

Table 1: Flooding scopes of Proxy-PAR registrations.

3 Proxy-PAR Protocols

3.1 Hello Protocol

The Proxy-PAR Hello Protocol is closely related to the Hello protocol specified in [2]. It uses the same packet header and version negotiation methods. For the sake of simplicity, states that are irrelevant to Proxy-PAR have been removed from the original PNNI Hello protocol. The purpose of the Proxy-PAR Hello protocol is to establish and maintain a Proxy-PAR adjacency between the client and server that supports the exchange of registration and query messages. If the protocol is executed across multiple, parallel links between the same server and client pair, individual registration and query sessions are associated with a specific link. It is the responsibility of the client and server to assign registration and query sessions to the various communication instances. Proxy-PAR can be run in the same granularity as ILMI [4] to support virtual links and VP tunnels.

In addition to the PNNI Hello, the Proxy-PAR Hellos travelling from the server to the client inform the client about the lifetime the server assigns to registered information. The client has to retrieve this interval from the Hello packet and set its refresh interval to a value below the obtained time interval in order to avoid the aging out of registered information by the server.

3.2 Registration/Query Protocol

The registration and query protocols enable the client to announce and learn about protocols supported by the clients. All query/register operations are initiated by the clients. The server never tries to push information to the client. It is the client’s responsibility to register and refresh the set of protocols supported and to re-register them when changes occur. In the same sense, the client must query the information from the server at appropriate time intervals if it wishes to obtain the latest information. It is important to note that neither client nor server is supposed to cache any state information about the information stored by the other side.

Registered information is associated with an ATM address and scope inside the PNNI hierarchy. From the IP point of view, all information is associated with a VPN ID, IP address, subnet mask, and IP protocol family. In this context, each VPN refers to a completely separated IP address space. For example <A, 194.194.1.01, 255.255.255.0, OSPF> describes an OSPF interface in VPN A. In addition to the IP scope further parameters can be registered that contain more detailed information about the protocol itself. In the above example this would be OSPF-specific information such as the area ID or router priority. However, Proxy-PAR server takes only the ATM and IP-specific information into account when retrieving information that was queried. Protocol specific information is never looked at by a Proxy-PAR server.

3.2.1 Registration Protocol

The registration protocol enables a client to register the protocols and services it supports. All protocols are associated with a specific AESA and membership scope in the PNNI hierarchy. As the default scope, implementations should choose the local scope of the PNNI peer group. In this way, manual configuration can be avoided unless information has to cross PNNI peer group boundaries. PNNI is responsible for the correct flooding either in the local peer group or across the hierarchy.

The registration protocol is aligned with the standard initial topology database exchange protocol used in link-state routing protocols as far as possible. It uses a window size of one. A single information element is registered at a time and must be acknowledged before a new registration packet can be sent. The protocol uses ' initialization' and 'more' bits in the same manner PNNI and OSPF do. Any registration on a link unconditionally overwrites all registration data previously received on the same link. By means of a return code the server indicates to the client whether the registration was successful.

Apart form the IP-related information, the protocol also offers a generic interface to the PNNI flooding. By means of so-called System Capabilities Information Groups other information can be distributed that can be used for proprietary or experimental implementations.

3.2.2 Query Protocol

The client uses the query protocol to obtain information about services registered by other clients. The client requests services registered within a specific membership scope, VPN and IP address prefix. It is always the client’s task to request information, the server never makes an attempt to push information to the client. If the client needs to filter the returned data based on service-specific information, such as BGP AS, it must parse and interpret the received information. The server never looks beyond the IP scope.

The more generic interface to the flooding is supported in a similar manner as the registration protocol.

4 Supported Protocols

Currently the protocols indicated in Table 2 have been included. Furthermore, for protocols marked ’yes’, additional information has been specified that is beneficial for their operation. Many of the protocols do not need additional information; it is sufficient to know they are supported and to which addresses they are bound.

To include other information in an experimental manner the generic information element can be used to carry such information.

5 VPN Support

To implement virtual private networks all information distributed via PAR can be scoped under a VPN ID [1]. Based on this ID, individual VPNs can be separated. Inside a certain VPN further distinctions can be made according to IP-address-related information and/or protocol type.

In most cases the best VPN support can be provided when Proxy-PAR is used between the client and server because in this way it is possible to hide the real PNNI topology from the client. The PAR capable server translates from the abstract membership scope into the real PNNI routing level. In this way the real PNNI topology is hidden from the client and the server can apply restrictions in the PNNI scope. The server can for instance have a mapping such that the membership scope "global" is mapped to the highest level peer group to which a particular VPN has access. Thus the membership scopes can be seen as hierarchical structuring inside a certain VPN. With such mappings a network provider can also change the mapping without having to reconfigure the clients.

For more secure VPN implementations it will also be necessary to implement VPN ID filters on the server side. In this way a client can be restricted to a certain set (typically one) of VPN IDs. The server will then allow queries and registrations only from the clients that are in the allowed VPNs. In this way it is possible to avoid an attached client from finding devices that are outside of its own VPN. There is even room for further restriction in terms of not allowing wildcard queries by a client. In terms of security, some of the protocols have their own methods, so PAR is only used for the discovery of the counterparts. For instance OSPF has an authentication that can be used during the OSPF operation. Hence even in the case where two wrong partners find each other, they will not communicate because they will not be able to authenticate each other.

Protocol	Additional Info
OSPF	yes
RIP	
RIPv2	
BGP3	
BGP4	yes
EGP	
IDPR	
MOSPF	yes
DVMRP	
CBT	
PIM-SM	
IGRP	
IS-IS	
ES-IS	
ICMP	
GGP	
BBN SPF IGP	
PIM-DM	
MARS	
NHRP	
ATMARP	
DHCP	
DNS	yes

Table 2: Additional protocol information carried in PAR and PPAR.

The VPN ID used by PAR and Proxy-PAR is aligned with the VPN ID used by other protocols from the ATM Forum and IETF. The VPN ID is structured into two parts, namely the 3-byte-long OUI plus a 4-byte index.

6 Interoperation with ILMI based Server Discovery

PAR can be used to complement the server discovery via ILMI as specified in [11,12,13]. It can be used to provide the flooding of information across the PNNI network. For this purpose a server has to register with a PAR-capable device. This can be achieved via Proxy-PAR or a direct PAR interaction. Manual configuration would also be possible. For instance the ATMARP server could register its service via Proxy-PAR. A direct interaction with PAR will be required in order to provide an appropriate flooding scope.

A PAR-capable device that has the additional MIB variables in the Service Registry MIB can set these variables when getting information via PAR. All required information is either contained in PAR or is static, such as the IP version.

7 Security Consideration

The Proxy-PAR protocol itself does not have its own security concepts. As PAR is an extension of PNNI, it has all the security features that come with PNNI. In addition, the protocol is mainly used for automatic discovery of peers for certain protocols. After the discovery process the security concepts of the individual protocol are used for the bring-up. As explained in the section about VPN support, the only security considerations are on the server side, where access filters for VPN IDs can be implemented and restrictive membership scope mappings can be configured.

8 Conclusion

This document describes the basic functions of Proxy-PAR, which has been specified within the ATM Forum body. The main purpose of the protocol is to provide automatic detection and configuration of non-ATM devices over an ATM cloud.

In the future, support for further protocols and address families may be added to widen the scope of applicability of Proxy-PAR.

9 Bibliography

[1] Fox, B. and B. Gleeson, "Virtual Private Networks Identifier", RFC 2685, September 1999.

[2] ATM-Forum, "Private Network-Network Interface Specification Version 1.0." ATM Forum af-pnni-0055.000, March 1996.

[3] ATM-Forum, "PNNI Augmented Routing (PAR) Version 1.0." ATM Forum af-ra-0104.000, January 1999.

[4] ATM-Forum, "Interim Local Management Interface, (ILMI) Specification 4.0." ATM Forum af-ilmi-0065.000, September 1996.

[5] Laubach, J., "Classical IP and ARP over ATM", RFC 2225, April 1998.

[6] Moy, J., "Extending OSPF to Support Demand Circuits", RFC 1793, April 1995.

[7] ATM-Forum, "LAN Emulation over ATM 1.0." ATM Forum af-lane-0021.000, January 1995.

[8] Armitage, G., "Support for Multicast over UNI 3.0/3.1 based ATM Networks", RFC 2022, November 1996.

[9] Droz, P., Haas, R. and T. Przygienda, "OSPF over ATM and Proxy PAR", RFC 2844, May 2000.

[10] Coltun, R., "The OSPF Opaque LSA Option", RFC 2328, July 1998.

[11] Davison, M., "ILMI-Based Server Discovery for ATMARP", RFC 2601, June 1999.

[12] Davison, M., "ILMI-Based Server Discovery for MARS", RFC 2602, June 1999.

[13] Davison, M., "ILMI-Based Server Discovery for NHRP", RFC 2603, June 1999.

Authors' Addresses

Patrick Droz
IBM Research
Zurich Research Laboratory
Saumerstrasse 4
8803 Ruschlikon
Switzerland

EMail: dro@zurich.ibm.com

Tony Przygienda
Siara Systems Incorporated
1195 Borregas Avenue
Sunnyvale, CA 94089
USA

EMail: prz@siara.com

Full Copyright Statement

Acknowledgement

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