Internet Engineering Task Force J. Palet
Internet-Draft M. Diaz
Expires: December 9, 2004 Consulintel
P. Savola
CSC/ FUNET
June 10, 2004

Analysis of IPv6 Tunnel End-point Discovery Mechanisms
draft-palet-v6ops-tun-auto-disc-01.txt

Status of this Memo

Copyright Notice

Abstract

Tunneling is commonly used in several IPv6 transition mechanisms. To be able to automate setting up tunnels, one critical component is being able to automatically determine the tunnel end-point for the tunneling mechanism. This memo analyses the different approaches for

configuring the IPv6 tunnel endpoint on a node.

Table of Contents

1. Introduction

Tunneling is commonly used in several IPv6 transition mechanisms. It is critically important to make setting up IPv6 connectivity simpler, so that it can be done simply also by IPv6-ignorant, novice users, or even completely transparently, without the user even having to know that IPv6 connectivity has been obtained.

One critical piece in the automated set-up is discovering the end-point for the IPv6-in-IPv4 (or possibly in the future, IPv4-in-IPv6) tunnel. Note that the other end-point ("tunnel server") typically also needs to have a means to configure the client's end-point, but that is assumed to be transition mechanism specific, and beyond the scope of this memo. A solution is being designed [1] based on the tunnel server/broker concept [2] which will, in particular, require this kind of discovery.

Many already-specified mechanisms already include a form of auto-discovery: for example, 6to4 [3] uses global anycast [4] and/or vendor's branch of DNS, Teredo [5] uses vendor's branch of DNS, and ISATAP [6] uses search-path -prefixed DNS.

2. Scenarios for Tunnel Endpoint Discovery

At least three scenarios can be identified where tunnel endpoint discovery would be useful.

2.1 Scenario 1: Initial IPv6 Deployment Stage

During the initial IPv6 deployment stage, the ISPs may not provide native IPv6 connectivity, at least in the access network. However, the ISP might offer IPv6 connectivity (probably for free) through an automatically set-up tunnel.

In this scenario, the users (or rather, their operating systems) need to be capable of automatically detecting whether the user's ISP is offering such service or not, and setting up the tunnel if available.

If this kind of IPv6 connectivity is set up automatically, it could create a load on the ISP's equipment which is configured as the tunnel-endpoint (e.g., a tunnel server). This is particularly important if state needs to be maintained. To address this consideration, the discovery method should allow for multiple end-points within a domain or even including a load-balancing mechanism.

2.2 Scenario 2: Initial IPv6 Support from External ISP

During the initial IPv6 deployment stage, the ISPs might not support IPv6 at all; there are thousands of ISPs, and many certainly won't be supporting IPv6 any time soon. The customers of those ISPs then have to use automatic tunneling mechanisms such as 6to4 or Teredo, or get a third-party ISP for IPv6 connectivity.

In this scenario, the users (in general their operating systems) may have a capability of automatically selecting a third-party ISP which is servicing outsiders. The service is often free of charge. The discovery process could either detect the closest serving end-point, or pick the one manually configured by the user.

Given the fact that IPv6 service could be offered by third parties, some kind of authentication could be required in order to allow only registered customers to use the IPv6 service. The authentication method will depend on the transition mechanism, so it is out of scope of this memo.

TBD: should this scenario be removed? Third party ISPs may be not economically feasible for free, even if there is some limited deployment of them at the moment. But it could be a registered and paid external service, for example a roaming service agreed among differnt ISPs.

2.3 Scenario 3: Nomadic Users

Nomadic users require connectivity to Internet from everywhere, from different locations: meetings, conferences, holidays, etc. Under this circumstance (always) obtaining native IPv6 connectivity is not feasible. The user has two choices: to discover a local tunnel (with different IPv6 addresses and prefixes) if provided by the local ISP, or to connect to the "home ISP" or "home network", implying the possibility of keeping the same addresses.

A local tunnel is typically a preferable choice, and could also be used as a Mobile IPv6 [7] care-of address. However, in many cases, the local ISP may not be providing a tunnel service.

Connecting to a "home provider" to obtain the tunnel is typically a safe choice, provided that the "home provider" allows IPv4 addresses outside its own domain to use its tunnel services; at the very least, typically this will require some sort of authentication. However, especially when roaming on a different continent as the home network, the latencies, etc., may be undesirable, so one might want to keep

this only as a backup option in case other approaches fail.

The whole process for having a new IPv6 tunnel with a new provider should be as transparent as possible in order to avoid that users need to manually re-register or change the configuration in their host. It would be desirable that the architecture enables the users to get connected and re-connected to the nearest tunnel end-point without manual intervention (for example when moving).

2.4 Scenario 4: Advanced IPv6 Deployment Stage

When the IPv6 deployment is in a more advanced stage, namely more users in more places looking for IPv6 connectivity, it is possible that ISPs providing IPv6 connectivity need to start a broader deployment. For a best IPv6 service, it is feasible that they increase the performance by using a tunnel end-point cluster geographically distributed to cover a country, etc. Furthermore they could offer the users only one of the methods proposed below for accessing the IPv6 connectivity. Each time users get IPv6 connectivity, they could use the same accesing method but they could be assigned to different tunnel end-point belonging the cluster.

Under this schema, some kind of load balancing could be required in order to distribute the load among the ISP resources.

In order to let all the candidate tunnel end-points to know the configuration of the previous user’s tunnels, some kind of tunnel management should be defined. However it is strongly dependant on the transition mechanism used, so it is out of the scope of this document.

In any case, as stated before, the whole process for obtaining a new IPv6 tunnel with a new TS should be as much transparent as possible in order to avoid that users need to manually re-register or change the configuration in their host. It would be desirable that the architecture makes the users get connected and re-connected to the nearest tunnel end-point without manual intervention.

3. Analysis of Solutions

Several possible solutions to discovering the tunnel end-point can be imagined; this section describes them in detail.

3.1 Shared-unicast -based Solutions

An "anycast" (shared-unicast by some terminology: see [8]) address identifies a group of hosts, usually server hosts. When a client sends a datagram to a shared-unicast address, it is delivered to one of the shared-unicast servers based on the routing topology and metrics.

There are two possible ways of using "anycast": as a global service -- where a shared-unicast prefix is the same for everyone, and advertised in the Inter-domain routing -- or as a local service, where the service provider is sharing one of its own addresses on multiple nodes for example for load-balancing or redundancy reasons.

Global "anycast" might be best applicable in scenario 2 -- to automatically discover the closest serving third party ISP. However, this raises the question of feasibility of that scenario. Local "anycast" can be combined with other solutions, described later, to seamlessly provide multiple tunnel end-points inside a single domain.

A packet to a shared-unicast address may end up being delivered to more than one node. In addition, there is no guarantee that two consecutive datagrams sent from the same host towards the same shared-unicast address are going to be delivered to the same node. However, when the routing topology is stable and metrics are well-designed, the packets are regularly delivered to the same nodes.

It is also possible to only use an "anycast" address only for the initial handshake, to establish a stable unicast address of the end-point and to perform some initial negotiation (an example of such is described in [9]).

3.2 Centralized Broker-based Solutions

Inside a single administrative domain, it would also be possible to deploy a centralized server or a "broker", which should know, probably in real-time, the status of all the associated end-points. Furthermore, it could, by using some means, redirect the users the correct end-points. This mechanism would still need another complementary approach to find the centralized broker.

This approach is highly assumptive of the tunneling set-up mechanism, and likely requires the implementation of lengthy redirection/ negotiation features. As such, its applicability is not further analyzed here.

Applying a centralized model over multiple administrative domains, e.g., having a single server for the whole Internet, would be administratively and management-wise unfeasible. Nevertheless, agreements between several domains could make possible sophisticated models

3.3 DNS-based Solutions

As DNS is globally deployed and easy to use, it could provide a

means
   for discovering the end-point address.

There are roughly three kinds of different approaches with DNS-based discovery:

1. "global name": the systems look up a globally unique name, like www.tunnel-server.net. -- this could be applicable with (unfeasible) global inter-domain broker or anycast-based solutions, and is therefore not considered at more length.

2. "vendor branch": the operating system vendors may provide a DNS record which is looked up (e.g., "6to4.windows.microsoft.com."), giving the vendor some control over already deployed systems. This is typically only feasible to configure a global anycast address, or provide the address of the vendor's own service, and is not applicable in a multi-vendor environment, and is not considered further.

3. "prefixing the search path" [10]: one could look up a service-specific special string, like "_tunnel-server", appended by the DNS search path, e.g., "isp.example.com", resulting in a query of "_tunnel-server.isp.example.com". This assumes that the DNS search path is provided by the ISP, and used by the user; this applies to most users (advanced users may have their own domainnames, own DNS servers, etc., but those could be expected to manually configure the end-point information). This is the most interesting approach, explored at more length below.

The special string could be more complex and make reference somehow to the transition mechanism it will accept, i.e. 6to4_tunnel-server, 6in4_tunnel-server, teredo_tunnel-server, any_tunnel-server and so on.

All of these approaches are typically coupled with a manual override option, which can be used by the knowledgeable users to look up different names or to specify the IP address completely.

3.3.1 Prefixing the DNS Search Path

Prefixing the search path bears a bit more analysis. There a couple of fundamental questions: where to store the records (i.e., the prefix to use, and what to do with the conflicts), and how to store the information (i.e., whether to use A/AAAA records, other other records).

There are at least three concrete possibilities for how to store the information:

1. "A/AAAA/CNAME records": one could use just the regular records for storing the end-point address or name. A drawback is that

there is a slightly higher probability of collision, depending on the service identifier used. The advantage is that it's very simple to implement and use. This also doesn't offer advanced load-balancing features, beyond those already provided by DNS round-robin techniques [11].

2. "SRV records": SRV records were created specifically for service discovery and load-balancing, mainly as a means to provide the users (also external users) knowledge of services within a domain. Quite this amount of unambiguity would not be needed if the service identifier is unique enough and only used internally.
A slight drawback is that SRV records require slightly more implementation and possibly more round-trips (if the results aren't cached).

3. "NAPTR records": NAPTR records provide even more flexibility than SRV records. The drawback is even more implementation and more round-trips. TBD: needs more text.

The question of where to store the information has a few tradeoffs, also depending on how the information is being stored. Using a commonly used name as a service identifier coupled with A/AAAA records would likely lead to false positives. On the other hand, if a a prefix like '_tunnel-server' would be chosen, it would be quite unprobable that conflicts would appear in practise. It is also worth remembering that the result of a false positive -- i.e., getting an address which is not a valid end-point -- is not necessarily a huge problem because it's only an indicator that such service didn't exist in the domain, as long as the tunneling mechanism can recover from that scenario.

Another consideration is the deployment of wildcard DNS records. If A/AAAA records were to be used, such records might create false positives quite easily. Fortunately, wildcards are commonly deployed only for MX records. A benefit of using SRV records is that they use an additional level in the zones, like: _tunnel-server._udp.isp.example.com." -- this would prevent a wildcard record "*.isp.example.com" from harassing the discovery of the end-point. XXX: needs to be checked.

3.4 DHCP-based Solutions

In most situations, the users receive the IPv4 information by means of an IPv4 DHCP server. Consequently, one of the parameters to be provided by the DHCP server could be the tunnel end-point address,

e.g., as described in [12].

This approach has several drawbacks:

- It requires upgrading the DHCP client/server implementations to support this feature.

- It is restricted to the local ISP. That is, it will not be effective if the local ISP doesn't provide this parameter. This could be also be an advantage considering that the this would only support the tunnels provided by the local ISP, which would probably be of good quality.

- It will not work if DHCP client is not used. DHCP is omitted especially in many dial-up scenarios, where only PPP is used; DHCP is not used in some (advanced) xDSL setups which use static routing. Also, some managed networks do not use DHCP. Still, in many cases, DHCP is used between a customer and the ISP.

- If a router is providing local DHCP information (e.g., an ADSL router), the tunnel end-point information would have to be automatically proxied to the "local DHCP", or manually configured on the router to propagate to the hosts in the case that the router is not activating the tunnel itself.

- It requires manual configuration/update of the ISP's DHCP servers when there are changes to the tunnel end-points, similar to updating DNS, NTP, etc. servers.

3.5 PPP-based Solutions

In the case of PPP-like connections, specific PPP parameters could be passed to the clients, as part of the AAA signaling process. This solution has the same drawbacks (and advantages) as indicated for the DHCP-based solution. Further, there has been resistance to making extensions to PPP (e.g., passing IPv6 prefix options), so it is an open question whether this information could be passed as a standardized PPP option at all.

3.6 Combined Solutions

There is a particularly interesting combination: DNS-lookups with a service identifier combined with the DNS search path (particularly with A/AAAA/CNAME records), and shared-unicast. DNS lookups can provide a local IP address (or addresses) for the end-points, and the local "anycast" approach can be used for load-balancing or adding more end-points to the system transparently so that every user uses the topologically closest end-point.

Similar approach to "anycasting" the end-point address obviously

also
   work for "anycast" and DHCP or PPP -based solutions.

4. Conclusions

DNS appears to be the simplest means to achieve end-point discovery; DHCP and PPP have drawbacks and due to many scenarios where only one of them is used, both the solutions would be needed. Inter-domain anycast model appears to be practically unfeasible even if it could work especially if "anycast" was only used for the unicast address discovery.

In the DNS, the records could be stored either in A/AAAA/CNAME or SRV records. The former appears to be slightly advantageous, while the latter is provably correct and offers slightly better load-balancing features, rather than a simple round-robin (and whatever may be obtained using e.g., anycast). Further analysis is still needed on the tradeoffs of these approaches.

Local anycasting techniques using a shared-unicast address (or addresses) appear to be the most practical means for redundancy and load-balancing.

5. Security Considerations

If the tunnel end-point discovery is done in an insecure fashion, so that an attacker could influence the discovery process, the attacker could be able to hijack all the IPv6 communications. This must be kept in mind when analyzing the different discovery solutions, and spelled out explicitly in the requirements if the threats are to be mitigated in tunneling mechanisms somehow (e.g., using a return routability procedures).

In particular, the potential weaknesses of DNS bear some consideration.

6. IANA Considerations

This document requests no action for IANA.

[[note to RFC-editor: this section can be removed upon publication.]]

7. Acknowledgements

This memo was written as a consequence of real experience using IPv6 when traveling, number of talks during IETF meetings and specially the work with the unmanaged, ISP and enterprise v6ops design teams. The authors would also like to acknowledge the European Commission support in the co-funding of the Euro6IX project, where this work is being developed.

8 Informative References

[1] Durand, A. and F. Parent, "Requirements for assisted tunneling", draft-durand-v6ops-assisted-tunneling-requirements-00 (work in progress), March 2004.

[2] Durand, A., Fasano, P., Guardini, I. and D. Lento, "IPv6 Tunnel Broker", RFC 3053, January 2001.

[3] Carpenter, B. and K. Moore, "Connection of IPv6 Domains via IPv4 Clouds", RFC 3056, February 2001.

[4] Huitema, C., "An Anycast Prefix for 6to4 Relay Routers", RFC 3068, June 2001.

[5] Huitema, C., "Teredo: Tunneling IPv6 over UDP through NATs", draft-huitema-v6ops-teredo-01 (work in progress), February 2004.

[6] Templin, F., Gleeson, T., Talwar, M. and D. Thaler, "Intra-Site Automatic Tunnel Addressing Protocol (ISATAP)", draft-ietf-ngtrans-isatap-22 (work in progress), May 2004.

[7] Johnson, D., Perkins, C. and J. Arkko, "Mobility Support in IPv6", draft-ietf-mobileip-ipv6-24 (work in progress), July 2003.

[8] Hagino, J. and K. Ettican, "An analysis of IPv6 anycast", draft-ietf-ipngwg-ipv6-anycast-analysis-02 (work in progress), June 2003.

[9] Thaler, D. and L. Vicisano, "IPv4 Automatic Multicast Without Explicit Tunnels (AMT)", draft-ietf-mboned-auto-multicast-02 (work in progress), February 2004.

[10] Faltstrom, P., "Design Choices When Expanding DNS", draft-ymbk-dns-choices-00 (work in progress), May 2004.

[11] Brisco, T., "DNS Support for Load Balancing", RFC 1794, April 1995.

[12] Kim, P. and S. Park, "DHCP Option for Configuring IPv6-in-IPv4 Tunnels", draft-daniel-dhc-ipv6in4-opt-03 (work in progress), April 2004.

[13] Nordmark, E. and R. Gilligan, "Basic Transition Mechanisms for IPv6 Hosts and Routers", draft-ietf-v6ops-mech-v2-02 (work in

progress), February 2004.

Authors' Addresses

Jordi Palet Martinez
Consulintel
San Jose Artesano, 1
Alcobendas - Madrid
E-28108 - Spain

Phone: +34 91 151 81 99
Fax: +34 91 151 81 98
EMail: jordi.palet@consulintel.es

Miguel Angel Diaz Fernandez
Consulintel
San Jose Artesano, 1
Alcobendas - Madrid
E-28108 - Spain

Phone: +34 91 151 81 99
Fax: +34 91 151 81 98
EMail: miguelangel.diaz@consulintel.es

Pekka Savola
CSC/FUNET

Espoo
Finland

EMail: psavola@funet.fi

Intellectual Property Statement

Disclaimer of Validity

Copyright Statement

Acknowledgment

Funding for the RFC Editor function is currently provided by the

Internet Society.