Behave WG
Internet-Draft
Intended status: Standards Track
Expires: September 13, 2012

T. Savolainen
Nokia
J. Korhonen
Nokia Siemens Networks
D. Wing
Cisco Systems

March 12, 2012

Discovery of IPv6 Prefix Used for IPv6 Address Synthesis
draft-ietf-behave-nat64-discovery-heuristic-07.txt

Abstract

This document describes a method for detecting presence of DNS64 and for learning IPv6 prefix used for protocol translation on an access network. The method depends on existence of a well-known IPv4-only domain name "ipv4only.arpa". The information learned enables nodes to perform local IPv6 address synthesis and to potentially avoid traversal through NAT64 on dual-stack accesses and multi-interface deployments.

Status of this Memo

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

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

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."

This Internet-Draft will expire on September 13, 2012.

Table of Contents

1. Introduction

As part of the transition to IPv6, NAT64 [RFC6146] and DNS64 [RFC6147] technologies will be utilized by some access networks to provide IPv4 connectivity for IPv6-only nodes [RFC6144]. The DNS64 utilizes IPv6 address synthesis to create local IPv6 presentations of peers having only IPv4 addresses, hence allowing DNS-using IPv6-only nodes to communicate with IPv4-only peers.

However, DNS64 cannot serve applications not using DNS, such as those receiving IPv4 address literals as referrals. Such applications could nevertheless be able to work through NAT64, provided they are able to create locally valid IPv6 presentations of peers' IPv4 addresses.

Additionally, DNS64 is not able to do IPv6 address synthesis for nodes running validating DNSSEC enabled DNS resolvers, but instead the synthesis must be done by the nodes themselves. In order to perform IPv6 synthesis nodes have to learn the IPv6 prefix(es) used on the access network for protocol translation. The prefixes, which may be Network Specific Prefixes (NSP) or Well-Known Prefixes (WKP) [RFC6052], are referred in this document as Pref64::/n [RFC6146].

This document describes a best effort method for applications and nodes to learn the information required to perform local IPv6 address synthesis. The IPv6 address synthesis procedure itself is out-of-scope of this document. An example application is a browser encountering IPv4 address literals in an IPv6-only access network. Another example is a node running validating security aware DNS resolver in an IPv6-only access network.

The knowledge of IPv6 address synthesis taking place may also be useful if DNS64 and NAT64 are present in dual-stack enabled access networks or if a node is multi-interfaced [RFC6418]. In such cases nodes may choose to prefer IPv4 or alternative network interface in order to avoid traversal through protocol translators.

It is important to notice that use of this approach will not result in as robust, secure, and good behaving system as an all-IPv6 system would be. Hence it is highly recommended to upgrade nodes' destinations to IPv6 and utilize the described method only as a short-term solution.

2. Requirements and Terminology

2.1. Requirements

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 [RFC2119].

2.2. Terminology

NAT64 FQDN: One or more fully qualified domain names for NAT64 protocol translator entity.

Pref64::/n: The IPv6 prefix used on IPv6 address synthesis [RFC6146].

Well-Known IPv4-only Name (WKN): a fully qualified domain name, "ipv4only.arpa", well-known to have only A record.

Well-Known IPv4 Address: an IPv4 address that is well-known and mapped to the well-known name.

3. Node Behavior

A node requiring information about presence of NAT64 and the Pref64::/n used for protocol translation SHALL send a DNS query for AAAA records of a well-known IPv4-only fully qualified domain name: "ipv4only.arpa". The node MAY also need to perform DNS query for the A record of the well-known name in order to learn what is the IPv4 address of the well-known name and if the A record even exists (see also Section 6 Exit Strategy). The node may perform this check in both IPv6-only and dual-stack access networks.

When sending AAAA query for the well-known name a node MUST set "Checking Disabled (CD)" bit to zero, as otherwise the DNS64 will not perform IPv6 address synthesis hence does not reveal the Pref64::/n used for protocol translation.

A DNS reply with one or more non-empty AAAA records indicates that the access network is utilizing IPv6 address synthesis. A node MUST look through all of the received AAAA records to collect one or more Pref64::/n. Pref64::/n list may include Well-Known Prefix 64: ff9b::/96 [RFC6052] or one or more Network-Specific Prefixes. In the case of NSPs the node SHALL search for the IPv4 address of the well-known name inside of the received IPv6 addresses to determine the used address format.

An IPv4 address of the well-known name should be found inside synthetic IPv6 address at some of the locations described in [RFC6052]. If the searched IPv4 address is not found on any of the

standard locations the network must be using different formatting. Developers may over time learn on IPv6 translated address formats that are extensions or alternatives to the standard formats. Developers MAY at that point add additional steps to the described discovery procedures. The additional steps are outside the scope of the present document.

The node should ensure a 32-bit IPv4 address value is present only once in an IPv6 address. In case another instance of the value is found inside the IPv6, the node shall repeat the search with another IPv4 address, if possible.

In the case only one Pref64::/n was present in the DNS response: a node shall use that Pref64::/n for both local synthesis and for detecting synthesis done by the DNS64 entity on the network.

In the case of more than one Pref64::/n were present in the DNS response: a node SHOULD use all of them when determining whether other received IPv6 addresses are synthetic. However, for selecting Pref64::/n for the local IPv6 address synthesis node MUST use the following prioritization order, of which purpose is to avoid use of Pref64::/n containing suffixes reserved for the future [RFC6052]:

1. Use NSP having /96 prefix

2. Use WKP prefix

3. Use longest available NSP prefix

In the case of NXDOMAIN response or an empty AAAA reply: the DNS64 is not available on the access network, network filtered the well-known query on purpose, or something went wrong in the DNS resolution. All unsuccessful cases result in unavailability of a node to perform local IPv6 address synthesis. The node MAY periodically resend AAAA query to check if DNS64 has become available. The node MAY also continue monitoring for DNS replies with IPv6 addresses constructed from WKP, in which case the node SHOULD use the WKP as if it were learned during the query for the well-known name.

To save Internet's resources, if possible, a node should perform Pref64::/n discovery only when needed (e.g. when local synthesis is required, cached reply timeouts, new network interface is started, and so forth). The node SHALL cache the replies it receives during Pref64::/n discovery procedure and it SHALL repeat the discovery process when one third of the Time-To-Live of the Well-Known Name's synthetic AAAA DNS response is remaining.

3.1. Secure Learning of Pref64::/n

If a node is using insecure channel between itself and DNS64, or DNS64 entity itself is untrusted, it is possible for an attacker to influence node's Pref64::/n detection procedures. This may result in denial-of-service, redirection, man-in-the-middle, or other attacks. To protect against these attacks the node SHOULD communicate with trusted DNS64 over secure channel or use DNSSEC. NAT64 operators SHOULD provide facilities for secure learning of Pref64::/n: a secure channel and/or DNSSEC protection.

3.1.1. DNSSEC Requirements for the Network

If the operator has chosen to support nodes performing Pref64::/n learning securely with DNSSEC, the operator of the NAT64 device MUST perform the following configurations.

1. Have one or more fully qualified domain names for the NAT64 translator entities (later referred as NAT64 FQDN). In the case of more than one Pref64::/n being used in a network, e.g. for load-balancing purposes, it is for network administrators to decide whether a single NAT64's fully qualified domain name maps to more than one Pref64::/n, or whether there will be dedicated NAT64 FQDN per Pref64::/n.

2. Each NAT64 FQDN MUST have one or more DNS AAAA resource records with each IPv6 address consisting of Pref64::/n and 0's for the elements after the actual prefix.

3. Each Pref64::/n MUST HAVE PTR record that points to corresponding NAT64 FQDN.

4. Sign the NAT64 FQDNs' AAAA and A records with DNSSEC.

3.1.2. Node Behavior

A node SHOULD prefer secure channel to talk to DNS64, whenever possible. In addition, a node that implements DNSSEC validating resolver MAY use the following procedure to secure discovery of the Pref64::/n.

1. Heuristically find out a Pref64::/n candidate by making AAAA query for the "ipv4only.arpa" by following the procedure in Section 3. This will return one or more AAAA resource records. For each of those AAAA resource records node wishes to use securely, the node performs the following steps.

2. Send DNS PTR query for the IPv6 address of the translator (for "ipv6.arpa"), using the Pref64::/n from the step 1 and zeroes for the elements after the actual prefix length. This will return the NAT64 FQDN.

3. The node SHOULD compare the domain of the NAT64 FQDN with node's list of trusted domains. The means for node to learn the trusted domains is implementation specific. If the node has no list of trusted domains, the node MAY query user whether the domain can be trusted. The node MAY remember the answer for future use. If the node has no trust for the domain, the discovery procedure is not secure and remaining steps described below are not needed.

4. Send DNS AAAA query for the NAT64 FQDN.

5. Verify the DNS AAAA response matches the address obtained in step 1. It is possible that the NAT64 FQDN maps to multiple AAAA records, in which case the node has to check if any of the responses matches to the address obtained in step 1. The node must ignore other responses and not to use those for local IPv6 address synthesis.

6. Perform DNSSEC validation of the DNS AAAA response.

After the node has successfully performed the above five steps, the node can consider Pref64::/n securely learned.

3.2. Connectivity Check

After learning Pref64::/n, the node SHOULD perform connectivity check to ensure the learned Pref64::/n is functional. It could be non-functional for a variety of reasons -- the discovery failed to work as expected, the IPv6 path to the NAT64 is down, the NAT64 is down, or the IPv4 path beyond the NAT64 is down.

There are two main approaches to determine if the learned Pref64::/n is functional. The first approach is to perform a dedicated connectivity check. The second approach is to simply attempt to use the learned Pref64::/n. Each approach has some trade-offs (e.g., additional network traffic or possible user-noticable delay), and implementations should carefully weight which approach is appropriate for their application and the network.

The node SHOULD use implementation specific connectivity check server, but if that is not possible a node MAY do a PTR query of the Pref64::/n that may return a hostname. The node then does an A query of that hostname, which may return zero, one or more A resource records pointing to connectivity check servers used by the network

operator. Negative response to PTR or A query means there are no connectivity check servers available. The operator of a NAT64 entity MAY assist nodes in their connectivity checks by mapping each NAT64 FQDN to one or more DNS A resource records with IPv4 address(es) pointing to connectivity check server(s).

In case of one or more connectivity check servers being available for use, the node chooses the first one preferring vendor specific servers, if multiple are available. The node MAY perform separate connectivity check by sending an ICMPv6 Echo Request to IPv6 address synthesized by combining discovered Pref64::/n with an IPv4 address of the server used for the connectivity check. This will test the IPv6 path to the NAT64, the NAT64's operation, and the IPv4 path all the way to the connectivity check server. Alternatively the node MAY utilize implementation specific connectivity check protocol. If no response is received for the ICMPv6 Echo Request, the node sends another ICMPv6 Echo Request, a second later. If still no response is received, it sends a third ICMPv6 Echo Request 3 seconds later. If an ICMPv6 Echo Response is received, the node knows the IPv6 path to the connectivity check server is functioning normally. If, after the three transmissions of the ICMPv6 Echo Request, no response is received, the node learns this Pref64::/n may not be functioning, and the node MAY choose a different NAT64 (if a different NAT64 is available), choose to alert the user, or proceed anyway hoping the problem is temporal or only with the connectivity check itself.

If no separate connectivity check is performed before local IPv6 address synthesis, a node may monitor success of connection attempts performed with locally synthetized IPv6 addresses. Based on success of these connections, and based on possible ICMPv6 error messages received (such as Destination Unreachable Message), the mode MAY cease to perform local address synthesis and MAY restart Pref64::/n discovery procedures.

3.2.1. No Connectivity Checks Against ipv4only.arpa

Clients MUST NOT send a connectivity check to the address returned in the ipv4only.arpa query. This is because, by design, no server will be operated on the Internet at that address as such. Similarly, network operators MUST NOT operate a server on that address. The reason this address isn't used for connectivity checks is that operators who neglect to operate a connectivity check server will allow that traffic towards the Internet where it will be dropped and cause a false negative connectivity check with the client (that is, the NAT64 is working fine, but the connectivity check fails because a server is not operating at ipv4-only.arpa on the Internet and a server is not operated by the NAT64 operator). Instead, for the connectivity check, an additional DNS resource record is looked up

and used for the connectivity check. This ensures that packets don't unnecessarily leak to the Internet and reduces the chance of a false negative connectivity check.

3.3. Alternative Domain Names

Some applications, operating systems, devices, or networks may find it advantageous to operate their own DNS infrastructure to perform a function similar to ipv4-only.arpa, but using a different resource record. The primary advantage is to ensure availability of the DNS infrastructure and ensure the proper configuration of the DNS record itself. DNS For example, a company named Example might have their application query ipv4-only.example.com. Other than the different DNS resource record being queried, the rest of the operations are anticipated to be identical to the steps described in this document.

4. Operational Considerations for Hosting the IPv4-Only Well-Known Name

The authoritative name server for the well-known name shall have DNS record Time-To-Live (TTL) set to a long value in order to improve effectiveness of DNS caching. The exact TTL value depends on availability time for the used public IPv4 address.

The domain serving the well-known name must be signed with DNSSEC. See also Security Considerations section.

It is expected that volumes for well-known name related queries are roughly SOMETHING, TBD. The infrastructure required to serve well-known name is SOMETHING, TBD.

5. DNS(64) Entity Considerations

DNS(64) servers MUST NOT interfere or perform special procedures for the queries related to the well-known name until the time has arrived for the exit strategy to be defined and deployed.

6. Exit Strategy

A day will come when this tool is no longer needed. At that point best suited techniques for implementing exit strategy will be documented.

A node SHOULD implement configuration knob for disabling the Pref64::/n discovery feature.

7. Security Considerations

The security considerations follow closely those of RFC6147 [RFC6147]. If an attacker manages to change the Pref64::/n node discovers, the traffic generated by the node will be delivered to altered destination. This can result in either a denial-of-service (DoS) attack (if the resulting IPv6 addresses are not assigned to any device), a flooding attack (if the resulting IPv6 addresses are assigned to devices that do not wish to receive the traffic), or an eavesdropping attack (in case the altered NSP is routed through the attacker).

The zone serving the well-known name has to be protected with DNSSEC, as otherwise it will be too attractive target for attackers who wish to alter nodes' Pref64::/n discovery procedures.

A node SHOULD implement validating DNSSEC resolver for validating the A response of the well-known name query. A node without validating DNSSEC resolver SHOULD request validation to be performed by the used recursive DNS server and use secure channel when communicating with the DNS64.

For the secure Pref64::/n discovery the access network SHOULD sign the NAT64 translator's fully qualified domain name. A node SHOULD use the algorithm described in Section 3.1 in order to securely learn the Pref64::/n.

Lastly, best mitigation action against Pref64::/n discovery attacks is to add IPv6 support for nodes' destinations and hence reduce need to perform local IPv6 address synthesis.

8. IANA Considerations

According to procedures described in RFC3172 this document requests IANA and IAB to reserve a second level domain from the .ARPA zone for the well-known domain name. The well-known domain name could be, for example, "ipv4only.arpa".

The well-known name also needs to map to one but preferably to two different public IPv4 addresses.

8.1. About the IPv4 Address for the Well-Known Name

The IPv4 address for the well-known name, if possible, should be chosen so that it is unlikely to appear more than once within an IPv6 address and also as easy as possible to find from within the synthetic IPv6 address. An address not listed in the Section 3 of

[RFC5735] is required as otherwise DNS64 entity may not perform AAAA record synthesis. The address does not have to be routable or allocated to any node, as no communications are initiated to the IPv4 address.

Allocating two IPv4 addresses would improve the heuristics in cases where the primary IPv4 address' bit pattern appears more than once in the synthetic IPv6 address (NSP prefix contains the same bit pattern as the IPv4 address).

If no well-known IPv4 address is statically allocated for this method, the heuristic requires sending additional A query to learn the IPv4 address that is sought inside the received IPv6 address. Without knowing IPv4 address it is impossible to determine address format used by DNS64.

9. Acknowledgements

Authors would like to thank Cameron Byrne, Christian Huitema, Washam Fan, Peter Koch, Stephan Lagerholm, Zhenqiang Li, Andrew Sullivan, and Dave Thaler, for significant improvement ideas and comments.

10. References

10.1. Normative References

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

[RFC5735] Cotton, M. and L. Vegoda, "Special Use IPv4 Addresses", BCP 153, RFC 5735, January 2010.

[RFC6052] Bao, C., Huitema, C., Bagnulo, M., Boucadair, M., and X. Li, "IPv6 Addressing of IPv4/IPv6 Translators", RFC 6052, October 2010.

[RFC6146] Bagnulo, M., Matthews, P., and I. van Beijnum, "Stateful NAT64: Network Address and Protocol Translation from IPv6 Clients to IPv4 Servers", RFC 6146, April 2011.

[RFC6147] Bagnulo, M., Sullivan, A., Matthews, P., and I. van Beijnum, "DNS64: DNS Extensions for Network Address Translation from IPv6 Clients to IPv4 Servers", RFC 6147, April 2011.

10.2. Informative References

[RFC6144] Baker, F., Li, X., Bao, C., and K. Yin, "Framework for IPv4/IPv6 Translation", RFC 6144, April 2011.

[RFC6418] Blanchet, M. and P. Seite, "Multiple Interfaces and Provisioning Domains Problem Statement", RFC 6418, November 2011.

Appendix A. Example of DNS Record Configuration

The following BIND-style examples illustrate how A and AAAA records could be configured by NAT64 operator.

The examples use Pref64::/n of 2001:db8::/96 and example.com domain.

The PTR record for reverse queries (Section 3.1.1 bullet 3):

```
$ORIGIN 0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.8.b.d.0.1.0.0.2.IP6.ARPA.
@         IN      SOA   ns1.example.com. hostmaster.example.com. (
                        2003080800 12h 15m 3w 2h)
          IN      NS    ns.example.com.

          IN      PTR   nat64.example.com.
```

If the example.com does not use DNSSEC, the following configuration file could be used. Please note the nat64.example.com has both AAAA record with the Pref64::/n and A record for the connectivity check (Section 3.1.1 bullet 2).

```
example.com.  IN SOA  ns.example.com. hostmaster.example.com. (
                      2002050501 ; serial
                      100        ; refresh (1 minute 40 seconds)
                      200        ; retry (3 minutes 20 seconds)
                      604800     ; expire (1 week)
                      100        ; minimum (1 minute 40 seconds)
                      )

example.com.  IN NS  ns.example.com.

nat64.example.com.
              IN AAAA  2001:db8:0:0:0:0:0 nat64.example.com.
              IN A  192.0.2.1
```

If the example.com does use DNSSEC, the following configuration file could be used for A and AAAA records:

```
   example.com.  IN SOA  ns.example.com. hostmaster.example.com. (
                          2002050501 ; serial
                          100        ; refresh (1 minute 40 seconds)
                          200        ; retry (3 minutes 20 seconds)
                          604800     ; expire (1 week)
                          100        ; minimum (1 minute 40 seconds)
                          )

   example.com.  IN RRSIG SOA  5 2 100 20090803071330 (
                          20090704071330 17000 example.com.
                          TVgWsNQvsFmeNHAeccGi7+UI7KwcE9TXPuSvmV9yyJwo
                          4FvHkxVC1H+98EtrmbR4c/XcdUzdfgn+q+lBqNsnbAit
                          xFERwPxzxbX0+yeCdHbBjHe7OuOc2Gc+CH6SbT2lKwVi
                          iEx3ySqqNoVScoUyhRdnPV2A1LV0yd9GtG9mI4w= )

   example.com.  IN NS  ns.example.com.
   example.com.  IN RRSIG NS  5 2 100 20090803071330 (
                          20090704071330 17000 example.com.
                          Xuw7saDDi6+5Z7SmtC7FC2npPOiE8F9qMR87eA0egG0I
                          B+xFx7pIogoVIDpOd1h3jqYivhblpCoDSBQb2oMbVy3B
                          SX5cF0r7Iu/xKP8XrV4DjNiugpa+NnhEIaRqG5uoPFbX
                          4cYT51yNq70I5mJvvajJu7UjmdHl26ZlnK33xps= )

   nat64.example.com.
                 IN AAAA  2001:db8:0:0:0:0:0 nat64.example.com.
                 IN RRSIG SOA  5 2 100 20090803071330 (
                          20090704071330 17000 example.net.
                          TVgWsNQvsFmeNHAeccGi7+UI7KwcE9TXPuSvmV9yyJwo
                          4FvHkxVC1H+98EtrmbR4c/XcdUzdfgn+q+lBqNsnbAit
                          xFERwPxzxbX0+yeCdHbBjHe7OuOc2Gc+CH6SbT2lKwVi
                          iEx3ySqqNoVScoUyhRdnPV2A1LV0yd9GtG9mI4w= )

   nat64.example.com.
                 IN A  192.0.2.1
```

Authors' Addresses

   Teemu Savolainen
   Nokia
   Hermiankatu 12 D
   FI-33720 Tampere
   Finland

   Email: teemu.savolainen@nokia.com

Jouni Korhonen
Nokia Siemens Networks
Linnoitustie 6
FI-02600 Espoo
Finland

Email: jouni.nospam@gmail.com

Dan Wing
Cisco Systems
170 West Tasman Drive
San Jose, California 95134
USA

Email: dwing@cisco.com