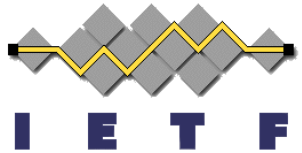
BEHAVE WG
IETF 79

Analysis of IPv6 address synthesis

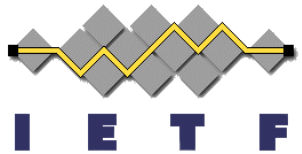
draft-korhonen-behave-nat64-learn-analysis-00



Thursday, November 11, 2010

Jouni Korhonen, Teemu Savolainen

Issues on IPv6 address synthesis



Issue #1) Finding out whether a particular IPv6 address is synthetic and therefore learning the presence of a NAT64 in the network.

Issue #2) Finding out how to construct from an IPv4 address an IPv6 address that will be routable to/by the NAT64.

Two Requirements.. (new)

1. Solution must support applications that do not use DNS.
2. Solution must support ability to learn multiple Pref64::/n.. (e.g. when DNS response contains multiple AAAAs and multiple Pref64::/n in one response).

- Blimey.. these, especially 2), came up late and would change the tone of the -00 analysis a bit..
  - Therefore, the above requirements are not reflected in the following presentation slides.

- We also need to understand the requirements properly, which applies, when and why.

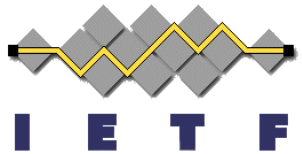
Background

- Certain applications, operating in protocol translation scenarios, can benefit from knowing the IPv6 prefix used by NAT64:
  - The Framework* document Scenario 1 and 5.

- DNS64 cannot serve applications that are not using DNS:
  - Such applications could still work through NAT64, provided they are able to create locally valid IPv6 presentations of peers' IPv4 addresses.

(*) https://datatracker.ietf.org/doc/draft-ietf-behave-v6v4-framework/

10 solution proposals for discovering IPv6 synthesis



1. EDNS0 option indicating AAAA Record synthesis and format.
2. EDNS0 flags indicating AAAA Record synthesis and format.
3. Heuristic discovery of NAT64 and Network-Specific Prefix.
4. DNS Resource Record for IPv4-Embedded IPv6 address.
5. Learning the IPv6 Prefix of a Network's NAT64 using DNS.
6. Learning the IPv6 Prefix of a Network's NAT64 using DHCPv6.
7. Learning the IPv6 Prefixes of an IPv6/IPv4 Translator (RA).
8. Using application layer protocols such as STUN.
9. Hybrid Type Prefix.
10. Provisioning of NAT64 Prefix and the format type.

Analysis on proposals

- See the I-D for detailed discussion on each proposal with listed PROs and CONs.
- In this slide set we only list the summaries for each proposal.

1. EDNS0 option indicating AAAA Record synthesis and format

- EDNS0 option based solution avoids the trouble of defining and standardizing a new DNS Resource Record type:
  - Extends the existing EDNS0 Resource Record.
- Solution has host resolver and DNS64 server impacts:
  - Only to the entities (end host, applications) that are interested in learning the presence of NAT64 and the used NAT64 prefix.
- The provisioning and management overhead is minimal, if not non-existent, as the EDNS0 options are synthesized in a DNS64 server in a same manner as the synthesized AAAA Resource Records.
- EDNS0 does not induce any load to DNS servers because no new RR Type query is defined.

Reference: draft-korhonen-edns0-synthesis-flag-01.txt

2. EDNS0 flags indicating AAAA Record synthesis and format

- Using the additional EDNS0 flags is included here for the sake of completeness.

- The consumption of three bits of the limited EDNS0 OPT space can be considered unfavorable and hence is unlikely to be accepted.

Reference: draft-korhonen-edns0-synthesis-flag-00.txt

3. Heuristic discovery of NAT64 and Network-Specific Prefix

- Uses a DNS query for an AAAA record of a (well-)known IPv4-only FQDN:
  - Possibly A query of the IPv4 address of the FQDN.
  - Finding IPv4 address within synthetic IPv6.
- Can be deployed without explicit support from the network or the host:
  - IPv4-only FQDN hosted by IANA(?), application vendor, host vendor, operator..
- This approach could also complement explicit methods and be used as a fallback approach when explicit methods are not supported by an access network.

Reference: draft-savolainen-heuristic-nat64-discovery-00.txt

4. DNS Resource Record for IPv4-Embedded IPv6 address (A6)

- Delivers explicit information about address synthesis taking place - solving the Issue #1.
- Standardization and deployment of a new DNS record type might turn out a too overwhelming task for a solution for a temporary transition phase.
- Defining a new record type increases load towards DNS server as the host issues parallel A64, AAAA and A queries.

Reference: draft-boucadair-behave-dns-a64-02

5. Learning the IPv6 Prefix of a Network's NAT64 using DNS

- The implementation of this solution requires some changes to the applications and resolvers – additional logic for TXT & U-NAPTR handling/information passing.

- Unlike the other DNS-based approaches, the U-NAPTR-based solution also requires provisioning information into the '.ip6.arpa.' tree, which is not anymore entirely internal to the provider hosting the NAT64/DNS64 service.

Reference: draft-wing-behave-learn-prefix-04

6. Learning the IPv6 Prefix of a Network's NAT64 using DHCPv6

- The DHCPv6-based solution would be a good solution in a sense it hooks into general IP configuration phase, allows easy updates when configuration information changes, and does not involve DNS in general.

- Use of DHCPv6 would require changes on both end host and network side DHCPv6 implementations.

- It is not obvious that all devices that need translation services would implement DHCPv6:
  - For example, cellular 3GPP networks do not mandate hosts or network to implement or deploy DHCPv6, and furthermore DHCPv6 is not even supported to configure end host IPv6 address [I-D.korhonen-v6ops-3gpp-eps].

Reference: draft-wing-behave-learn-prefix-04

7. Learning the IPv6 Prefixes of an IPv6/IPv4 Translator (RA)

- The RA-based solution would be a good solution in a sense it hooks into general IP configuration phase, allows easy updates when configuration information changes and does not involve DNS in general.

- Introducing any changes to the Neighbor Discovery Protocol are not generally favored due the impact on both network node side and end host IP stack implementations.

- Compared to the DHCPv6 equivalent solution the management overhead is greater with RA-based solution:
  - In case of DHCPv6-based solution the management can be centralized to few DHCPv6 servers compared to RA-based solution where each access router is supposed to be configured with the same information.

Reference: draft-wing-behave-learn-prefix-03

8. Using application layer protocols such as STUN

- The STUN, or similar, protocol based approach is another way to solve the problem without explicit access network support.
- The heuristics for NSP discovery would still be in the client, however, the result may be more reliable as actual IPv4 destination address is compared to IPv6 address used in sending:
  - Client knows the IPv6 address it used, and it receives in response the IPv4 address STUN server received the packet on.
- The additional benefit of STUN is that the client learns its public IPv4 address with the same message exchange.
- STUN could also be used as the connectivity test tool if the client would first heuristically determine NSP, synthesize IPv6 representation of STUN server's IPv4 address, and then tests connectivity to the STUN server.

Reference: http://www.ietf.org/mail-archive/web/behave/current/msg08898.html and offline discussions

9. Hybrid Type Prefix

- This solution mitigates both Issue #1 and Issue #2, but it requires a new IPv6 address registry to be maintained by IANA. Two IPv6 prefix blocks need to be maintained by registries.

- To ensure global reachability in Internet, an additional announcement (HTP) is required to be advertised in the inter-domain routing.

Reference: draft-xu-behave-hybrid-type-prefix-00

10. Provisioning of NAT64 Prefix and the format type

- This DHCPv6-based solution allows a given IPv6 host to learn whether an IPv6 address is IPv6-Converted or not only if the NAT64 is managed by the same service provider as the one offering the connectivity service to the IPv6 host.
- The solution requires a new DHCPv6 option code to be assigned. This solution can be considered as a complementary solution to DNS-based ones.
- This proposal is designed for IPv4-in-IPv6 encapsulation uses, but could be used also with NAT64.
- Shares same concerns as proposal 6.

Reference: draft-boucadair-dhcpv6-shared-address-option-01

Conclusion of the analysis

- Using DNS to carry the additional synthetisation information feels logical:
  - The fact that the synthetisation information fate-shares the information received in the DNS response is a valuable attribute and reduces the possible distribution of stale prefix information.
- Use of DHCPv6 would require upgraded software (if usable at all) and requires synchronization between NAT64/DNS64 and DHCPv6 servers.
- RA-based mechanisms are hard to get deployed and operationally expensive.
- HTP needs a new IPv6 address registry from IANA, which sounds an overkill.
- The two heuristic-based approaches could be deployed easily without dependency to host operating system or networks:
  - Long term their usefulness depends on how well networks will deploy explicit methods.
  - STUN could be extended, or some other protocol used.

Summary & author's recommendations

- 10 solution proposals on the plate for the same problem. i.e. there is an issue to solve.

- Proposal:

1. Standardize ENDS0 option:
   - Lightweight implementation, standardization and management/provisioning.

2. Combine the two heuristic-based methods into one informational document for application and host implementers.

3. Publish this I-D as an Informational RFC?

Feedback?