Project Number:	**IST-2001-32603**
Project Title:	**6NET**
CEC Deliverable Number:	**32603/invenia/DS/3.2.2/A1**
Contractual Date of Delivery to the CEC:	30 April 2003
Actual Date of Delivery to the CEC:	1 August 2003
Title of Deliverable:	Proxy DNS Installed
Work package contributing to Deliverable:	WP3
Type of Deliverable*:	P
Deliverable Security Class**:	PU
Editors:	Feike W. Dillema
Contributors:	Christian Strauf, Gunter Van de Velde, WP2, WP3

* Type: P - Prototype, R - Report, D - Demonstrator, O - Other

** Security Class: PU- Public, PP – Restricted to other programme participants (including the Commission), RE – Restricted to a group defined by the consortium (including the Commission), CO – Confidential, only for members of the consortium (including the Commission)

Abstract:

This document describes the DNS proxy `totd' that is being developed by the 6NET project to provide DNS support for a variety of IPv4/IPv6 transitioning mechanisms. The proxy is described in general first. Then the special mechanisms it supports are described one at a time. After that, different uses of the proxy in a network are presented. Finally, the most important items left to add to the DNS proxy are discussed briefly.

Keywords:

DNS proxy, address translation, IPv4 IPv6 transitioning, 6to4, scoped addresses.

Executive Summary

This Deliverable formally represents the installation of the proxy DNS in the 6NET network. It provides general information concerning the reasons for applying a Proxy DNS for situations in which both IPv4 and IPv6 addresses have to be resolved. It describes the technique used, and the considerations to be made when installing a proxy DNS in a network. It also mentions 2 enhancements that will be tested in the project.

Table of Contents

1. INTRODUCTION 4

2. THE DNS PROXY `TOTD' 5

3. DNS RECORD TRANSLATION AND TRICKS 6

3.1. NAT-PT AND TRT 6
3.2. 6TO4 REVERSE LOOKUP 6
3.3. SCOPED ADDRESS REWRITE 7

4. DNS PROXY PLACEMENT AND CONFIGURATION IN A NETWORK 9

5. FUTURE WORK 10

6. CONCLUSIONS 11

1. Introduction

IPv4 and IPv6 will need to interoperate for many years. An important part of the 6NET project is therefore to investigate deployment of transition mechanisms. A number of transition mechanisms [RFC2766,RFC3142] operating at the network and transport level make use of IPv6 addresses that embed an IPv4 address. Applications and their users typically (should) use DNS hostnames to denote communication endpoints, instead of dealing with IP addresses directly. Expecting the users and applications to embed an IPv4 address into an IPv6 address is therefore unreasonable as they will expect the DNS system to map hostnames to whatever address(es) is/are usable for communication. Hence, we need support in the DNS for the generation of these special addresses needed for the transition mechanisms in question.

Address mapping support could be added in the DNS resolvers and/or servers, or a special-purpose DNS proxy could be used. The latter approach has been further developed as part of the 6NET project in the form of the IPv6/IPv4 translating DNS proxy `totd'. This document further describes this proxy, why it is needed, how it can and should be used and deployed and installed in networks in general and the 6NET network in particular.

2. The DNS proxy `totd'

Totd is just a DNS-proxy; it does not recursively resolve queries itself. It can only forward queries to one or more real nameservers that will recursively resolve queries. Such nameservers are called forwarders in `totd terminology'. If there are multiple forwarders specified, it will try them in the order listed. As a proxy totd `sits between' client resolvers and real forwarder nameservers and as such receives requests from clients which totd normally forwards to a real nameserver to resolve. When it subsequently receives a response from the nameserver it simply forwards it back to the client.

When a nameserver is or becomes unreachable totd will use the next nameserver/forwarder in line. After the retry interval amount of time, totd will switch back to the previous nameserver. If that nameserver is still unreachable it uses the next nameserver in the config file again. You may see totd switch to backup nameservers for no apparent good reason, as totd is not very good at discriminating between an unreachable/malfunctioning nameserver or a single query that returns erroneous results or is simply delayed a lot. (If the second nameserver is also unreachable and a third is specified, this sequence repeats itself.)

Such DNS proxy functionality with transport support for both IPv4 and IPv6 is already quite useful. However, the main reason for totd's existence is its ability to perform a number of translation tricks on the DNS queries and responses that it forwards. These tricks are the subject of the following section.

3. DNS Record Translations and Tricks

3.1. NAT-PT and TRT

The totd DNS proxy can treat each AAAA and A6 type query in a special way. This behavior is enabled when one or more prefixes are configured (on the commandline or with the `prefix' keyword in the config file). It is meant to support network and transport level IPv6 to IPv4 transition mechanisms, like NAT-PT [RFC2766] and transport level relays [RFC3142] like faith and TRT.

If the nameserver does not return an IPv6 address for the forwarded AAAA/A6 query, totd will make a second query but this time for an A record of the hostname of the original query. The resulting IPv4 address is then used to construct a fake IPv6 address, by replacing the lower 32 bits of the specified prefix with this IPv4 address. The resulting IPv6 address is sent as response to the original AAAA/A6 record query. In addition, totd treats PTR type queries (reverse name lookup) in the ip6.int. and ip6.arpa. domains specially. If the query matches a specified prefix, totd will forward a PTR query for an IPv4 address (using the lower 32 bits of the address to construct an IPv4 address) instead and use that to construct a faked response to the original PTR query.

If multiple prefixed are configured, totd will cycle through them in round-robin fashion. In this way totd can balance the load for multiple NAPT-PT/faithd/pTRTd translators in a network and support to some extent redundancy in the setup of transitioning mechanisms in a network.

Currently, there is no way to notify a running totd that a prefix is `out of order' and should not be used (except for stopping and restarting totd with a new configuration). Also, totd has no way to check itself that the prefixes it uses are actually `live', i.e. can be used to contact remote (IPv4-only) machines). 6NET project participants have requested such functionality and its implementation is planned for a future release of totd. This would provide proper support for failover and/or (IPv4) multihoming in totd.

3.2. 6to4 reverse lookup

The IETF draft [6to4dns3] discusses several solutions to the problem of reverse address resolution of 6to4 [RFC3056] addresses. As 6to4 addresses are IPv6 addresses that are derived directly from an IPv4 address and are NOT allocated or assigned by an IPv6 address registrar. An owner of an IPv4 address can start using the IPv6 address space starting with 2002, followed by the IPv4 address (converted to hexadecimal) without filing a request to a address registrar. As it is normally the task of the registrars to delegate the reverse address zone for the address space they allocate to address requesters, there is no (root to) the DNS tree for reverse lookup of 6to4 addresses.

One way to `solve' this is to assume that the reverse DNS for the 6to4 addresses is in the same hands as the reverse DNS for the IPv4 address used to derive them. This will often not be true. Consider for example a home user that is assigned an IPv4 address by their ISP that keeps control over its reverse DNS entry. The user derives its own 6to4 address space as his ISP provides no IPv6 service whatsoever and his ISP is deaf for requests to enter reverse DNS for the 6to4 addresses of the user. In such cases, the reverse DNS for the IPv4 address is under control of a different entity (the ISP) than the reverse DNS of the 6to4 addresses (the home user, if any). For that reason, the author of [6to4dns3] proposes another method that does work in such a case. This method could be supported by totd, but it currently is not. However, one can argue that for many, if not all, cases (like serverfarms) where the existence of reverse DNS is important, the reverse DNS of the IPv4

addresses will be under control of the same entity as the reverse DNS of the 6to4 addresses derived from them.

For the cases where the assumption holds, the totd proxy can be configured to make use of it in order to resolve 6to4 addresses without the existence of a root server for the [x2002].IP6.ARPA. domain. This functionality is experimental and optional as it may easily be obsoleted by the introduction of a root nameserver for the 6to4 reverse address domain. Note, however, that even if (or when) such a root server exists, configuring totd to do the special 6to4 reverse lookup will not produce wrong results (the root server based DNS tree takes precedence).

The following explanation of the special reverse lookup for 6to4 is based on the text of section 3.3 of [6to4dns0]. When the proxy receives a PTR or NS query for a label that has a [x2002].IP6.ARPA suffix, it would first attempt to satisfy that query forwarding the query to an upstream server (the forwarder nameserver). If that query failed due to a "no such domain" error, the proxy then attempts to find the server for the {some-thing}.[x2002].IP6.ARPA label by issuing an NS query for {something}.IN-ADDR.ARPA.

If the original query is for PTR records, and one or more NS records are found for {something}.IN-ADDR.ARPA, the proxy then forwards the original query for {something}.[x2002].IP6.ARPA to one or more of those servers, and returns the results from the first successful one of the forwarded queries, if any.

If the original query is for NS records, and one or more NS records are found for {something}.IN-ADDR.ARPA, the proxy then returns the pseudo-records corresponding to the IN-ADDR.ARPA domains. Those pseudo-records are NOT marked as authoritative. Even though, in the above, IP6.ARPA. has been used throughout, totd's 6to4 reverse lookup support works the same for the IP6.INT. tree.

3.3. Scoped address rewrite

Totd supports rewriting of scoped addresses in DNS responses. This technique allows usage of scoped addresses like site-local addresses, without having to maintain such addresses in a DNS database. Instead, totd derives the scoped address record from the global address record it does find in DNS. This is especially useful for sites where all machines have site-local addresses, and the use of site-local addresses is preferred over global ones (as they are immune to prefix renumbering events for example). This optional and experimental functionality in totd is based on requests and contributions from the Kame project [kame].

Totd only performs this trick on queries that stay within the specified scope. I.e. if the query is made from scoped source address (link-local unicast or site-local unicast), and the query's target address (totd's listening address) is also scoped address, totd attaches additional AAAA records converted by using 3 arguments of the scoped keyword.

When you configure as below in totd config file:

```
scoped 3ffe:ffff:ffff:: fec0:: 48
```

and you made query from scoped source to totd's scoped destination, and the result has the following record:

```
foo.kame.net.   IN AAAA 3ffe:ffff:ffff::9876:5432
```

the response from totd will get additional records as follows:

foo.kame.net. IN AAAA 3ffe:ffff:ffff::9876:5432

foo.kame.net. IN AAAA fec0::9876:5432

Reverse query for fec0::9876:5432 will be converted into 3ffe:ffff:ffff::9876:5432 and forwarded to the real DNS servers (the forwarder in totd terminology).

4. DNS Proxy Placement and Configuration in a Network

Roughly, the totd DNS proxy can be placed in three ways:

a) On each single host. Running totd on a single host only for applications running on that host can especially be useful for laptop and other portables that roam between IPv6-only, dual-stack and IPv4-only networks and use IPv4-only and IPv6/IPv4 applications. This facilitates applications with an IPv4-only resolver to make DNS queries via the proxy when only IPv6-only DNS servers are available (the proxy receives then queries over IPv4 locally and forwards them over IPv6 to the nameserver).

Using totd on the own machine may also be more efficient when visiting a remote IPv6-only network, as it allows a user to use the local nameservers of the network he visits, while still using the IPv6 to IPv4 transition mechanism (and associated prefix) of the home network. This is likely to be inefficient (routing wise) in case the remote network also offers IPv4 connectivity either natively or by some transition mechanisms, but the convenience of not having to reconfigure may outweigh this. In this context, it is useful to note that it can be useful to define a standard DHCP option to configure prefixes of transition mechanisms like NAT-PT and TRT, such that reconfiguring becomes automatic.

b) At the upstreams (IPv6-only) provider. In this case, a user, host or site does not run its own nameservers (except for maybe a caching forward-only one), and the upstreams would provide some translation service between IPv6 and IPv4 like NAT-PT to its clients and with it the DNS proxy service to make use of it. This may likely become the common case in the future for ISPs for homeusers that phase out native IPv4 transport in their access networks to simplify network management.

c) One for a site. Running a single DNS proxy on a site allows all clients to be configured with the same (proxy) DNS server which does not need to change when the real nameservers (forwarders) change IP address (due to migration from IPv4 to IPv6 or due to a change in upstream network and DNS service provider, for example).

Two main cases can be discriminated:

- A site that has both IPv4 and IPv6 upstreams connectivity to run its own translation mechanism for its IPv6-only hosts to connect to the IPv4 world.

- A site that relies upon the translation mechanism(s) in place at his upstreams network provider, but likes to be able to quickly switch to a translator elsewhere. Examples are multihomed sites, or sites that a backup upstreams that is only called upon after the regular upstreams has failed.

d) One in an internetwork like 6NET. Running a totd DNS proxy shared by multiple sites of an internetwork can be useful for testing transition mechanisms deployed somewhere in the network. Sharing the DNS may then be an easy way to redirect the traffic of multiple sites to the mechanism under test such that realistic and/or extreme test loads can be produced.

One could also share the translation mechanisms of sites between sites to achieve redundancy/multihoming of IPv4 traffic via the different IPv4 exit/gateway points of the IPv6-only network. However, totd currently has little support for efficient failover as described earlier. This may change, if (6NET) users deem such support desirable.

5. Future Work

Based on feedback from 6NET participants work is planned on implementing one or two control mechanisms to notify a DNS proxy of operational changes like a prefix that should not be used anymore or a new prefix that it should start using. We plan to implement two mechanisms; one that is fairly easy to implement such that it can be introduced quickly and later a second one that requires more in design and implementation but gives more flexibility.

Here follows a short description of the two mechanisms:

1. Implement a control client (e.g. "totd-ctrl") that can send commands to a running totd-instance locally on the DNS-proxy host.

For example:

"totd-ctrl --deactivate fec0:0:0:ffff::"

and:

"totd-ctrl --reactivate fec0:0:0:ffff::"

This would, for example, allow a simple command sequence like this:

ping6 -c 1 -w 2 fec0:0:0:ffff::1 >/dev/null 2>&1 || totd-ctrl –deactivate fec0:0:0:ffff::

to make totd stop using the prefix of a TRTd server that has become unreachable.

2. Let totd listen on a control-port so that one can send control messages to it with some kind of remote client. With this feature, one could either implement "dead-man" controls. For example, a TRT-server can then be forced to use the control client to periodically send a certain control string -- if it fails to send this control string, some specific prefix will not be used for IPv4 address conversion anymore) or one could use it from a surveillance-server that probes TRT-servers periodically.

Finally, we plan to support the mechanisms described in section 3.2 of [6to4dns3].

6. Conclusions

This Deliverable has formally represented the installation of the proxy DNS in the 6NET network. It provided general information concerning the reasons for applying a Proxy DNS for situations in which both IPv4 and IPv6 addresses have to be resolved. It described the technique used, and the considerations to be made when installing a proxy DNS in a network. It also mentioned some future enhancements that will be tested in the project.