IP-FF Babysitter: Stateful Network Address Translation
=================================================================
including Port, Protocol and Domain Name Translation
for Internet Protocol - Five Fields
Specification Draft

Abstract

Babysitter is a form of an advanced NAT, mostly for desktop clients. It gives mixed IP-FF and IPv4 clients access to IPv4-only Internet. It is somewhat resembling NAT64 + DNS64 combo, and will aid during transition period.

Assumption: We work on IPv4-only Internet, but we want to implement both IP-FF and IPv4 hosts inside our organization, so nodes can work between themselves with, and take advantage of, IP-FF, but still able to connect to the Internet.
If/when this assumption is invalid, and end-to-end IP-FF becomes commonplace, other forms of connection should be used, and babysitter may be disabled.

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."

Copyright Notice

Copyright (c) 2015 IETF Trust and the persons identified as the document authors. All rights reserved.
Table of Contents:

1. IP-FF Babysitter components
2. IP-FF Babysitter requirements
3. Address mapping
4. DHCP default settings
5. DNS ALG translation: conceptual workflow
6. NAT table: Logical construction and work flow
7. Checksums
8. Limitations
9. Load-balancing multiple babysitters in parallel, for scalability
10. Duplicate Address Detection proxy

1. IP-FF Babysitter components

Stateful NAT and DHCP (client and server), Stateless DNS ALG and Address mapping.

2. IP-FF Babysitter requirements

IP-FF babysitter, at a minimum, should support stateful ICMP echo, as well as TCP and UDP protocols and DNS translation by ALG (Application-Level Gateway)

Supporting other ICMP commands and transport protocols is a bonus, as are Application-Level Gateways (ALGs) for poorly-behaving protocols.
Port redirection is a bonus, too. (so that incoming sessions are made possible on specific ports)

IP-FF Babysitter takes only one IPv4 address, either public or private, and MUST work even behind CGN (Carrier-grade NAT), where a private IPv4 address is supplied via DHCP.

3. Address mapping

Source IP addresses are translated in many-to-one fashion.
Destination IP addresses are mapped as a simple one-to-one function.

IPv4:   a.b.c.d

becomes

IPFF: 10.a.b.c.d
Visually similar!

And if you need private addresses?

10.10.x.x/20 - all yours!

When address is mapped, no DAD is performed, since multiple Babysitters can exist on the same network segment.

4. DHCP default settings

10.10.0.5.999/40 = Default Gateway (babysitter itself)
Typically it should give it’s DHCPv5 clients the range between 10.10.0.5.10-990/40

DNS Server’s IP may be mapped to whatever DNS address is provided by your Internet Service Provider (ISP).

i.e. if your ISP gives you DNS = 82.102.139.10
Babysitter maps it as 10.82.102.139.10, and gives it to clients via DHCP reply.

Alternatively, Babysitter MAY implement a full DNS proxy with caching.

If IPFF-babysitter is a DHCP (v4) client itself, DHCP-FF address leases to clients should be a bit shorter or equal to what this babysitter itself receives.

Additionally, to support IPv4 nodes, Babysitter includes a DHCPv4 server.

10.0.5.254/24 = Default Gateway (babysitter itself)
Typically it should give it’s DHCPv4 clients the range between 10.0.5.10-250/24

Any or both of DHCPv4 or DHCPv5 servers can be disabled, or re-configured.

5. DNS ALG translation: conceptual workflow

This is conceptually similar to DNS64, where Babysitter translates DNS queries on the fly.
The NAT itself synthesizing AA records from A records.

IPFF-Babysitter DNS, unlike DNS64 or NAT-PT, does not check if companyABC.com supports IPFF "AA" resource record (RR) or not, but looks only for "A" Resource Records.

DNS Request:

+--------+ +-----------------+ +-----------------+
| IPFF node | IPFF-Babysitter | IPv4 DNS Server |
+--------+ +-----------------+ +-----------------+
        |  |  |  |
--->  "AA" query  "A" query
        companyABC.com  companyABC.com
          (step 1)            (step 2)
DNS Reply:

+---------+   +---------------+   +---------------+
|IPFF node|---|IPFF-Babysitter|---|IPv4 DNS Server|
+---------+   +---------------+   +---------------+

<--                 <--
"AA" reply            "A" reply
10.28.211.136.15       28.211.136.15
= companyABC.com       = companyABC.com
(step 4)               (step 3)

DNS translation should be stateless, but in order to prevent translation of "A" responses, sent from dual-stack or IPv4 clients, it should look at the stateful UDP NAT table, and ONLY if the client is IP-FF node (DNS query via IP-FF Transport), then translate DNS response to "AA" record. If the client is IPv4 node (DNS query via IPv4 Transport), no DNS ALG translation is needed.

6. NAT table: Logical construction and work flow

Assuming our Babysitter is itself behind IPv4 NAT, and got a private IPv4 address of 10.0.0.4. This will be our translated source IP. And it has an IP-FF client, that got an address 10.10.0.5.10.

| IPv4 node|----------|
| 10.10.0.5.x/24 |
+---------+
 .254

10.10.0.5.x/40 | 10.0.0.x/24 | 4.5.6.x/24
.10 .999 | .4 .254 .7 .8

IPFF node---IPFF-Babysitter---IPv4 NAT---IPv4 Web Server
traffic to web server traffic to web server
from 10.10.0.5.10 from 10.0.0.4
to 10.4.5.6.8 to 4.5.6.8

For outbound packets, source NAT is always stateful, and port translating.

original.src.IP|translated.src.IP|original.src.port|translated.src.port
-----------------------------------------------
10.10.0.5.10 | 10.0.0.4 | TCP:1027 | TCP:2031
10.0.5.10 | 10.0.0.4 | TCP:1027 | TCP:2032

NOTE: "src." = Source

Source NAT table must be in mixed IP-FF & IPv4 format, due to the possibility to serve both legacy IPv4 nodes, as well as newer IP-FF nodes. Translated source address is always a single IPv4 address. Since there is only one IPv4 address in the whole IPFF babysitter, so translated source IP can be only it.
This single IPv4 address can change if, for example, Babysitter disconnected from 4G/LTE network and went Ethernet or ADSL. IP-FF Babysitter must be able to change it’s IPv4 address mapping on the fly. (existing connections may break, but new can be established)

Very similar to what NAT devices have now, simplified here.

Variety of clients:

Assume we have both IPv4 and IP-FF clients. For IPv4 clients, Babysitter acts like a typical NAT (NAPT) does. We must use a single NAT table, to avoid duplicate translated source port. NAT adds IPv4 nodes to it’s own NAT table, but keep original source address mixed.

7. Checksums

Checksums must be recomputed when dealing with address translation, as the IP pseudo-header is always different. Checksums must be computed according to the rules of each Address Family and protocol.

8. Limitations

Obviously serving content is no joy via a Babysitter. Specific TCP/UDP port forwarding will need to be done manually.

"Well-behaved" applications will work great, as they do over standard NAPT. Some may break. But web browsing (HTTP, HTTPS) will work.

Standard set of limitations, applied to the NAPT applies to Babysitter also.

Babysitter is pretty much incompatible with First Field Overlap (FFlap), as defined in "LARA" spec, since no DAD is performed on mapped addresses.

9. Load-balancing multiple babysitters in parallel, for scalability

In cases, where one Babysitter can’t handle the workload, either due to single IPv4 limitation, or CPU processing limitation, just hard-limit the amount of DHCP addresses leased to something small. Perhaps 30-50 addresses.

Add another Babysitter, and because of Duplicate Address Detection, it will find it’s own address quickly, and clients will find their own, and if there is a duplicate IP (collision), clients will request the next IP address.

Babysitter always gives it’s own IP as a default gateway via DHCP. So 2nd babysitter will give itself the next available address in the same subnet, something like : 10.10.0.5.998 This allows for per-client load-balancing.
DHCP server inside Babysitter should to be smart. That is, it should artificially *delay* giving IP-FF addresses, if CPU usage is high, allowing for another Babysitter to answer DHCP, become a gateway, a NAT and a DNS ALG resolver.

For DHCP server throttling, see [IPFF-DHCP], Section 4.

10. Duplicate Address Detection proxy

When Babysitter exists, it SHOULD answer all DAD requests as a DAD proxy, to (at least inform or ) prevent hosts from taking any possible mapped-IPv4 public address, even if unused.

I.e. it should mark all the mapped public IPv4 addresses as "used".

For example some node sends a DAD request to check for 10.1.1.1, since 1.1.1.1 is a valid public IPv4 address, reserved for mapping, so Babysitter should send a DAD reply indicating this address is used.

But if someone sends a DAD request to check for 10.10.0.0.1, which may be a mapped private IPv4 address, Babysitter should not answer (except if used by Babysitter itself).

NOTE: DAD is defined in [IPFF-LARA] spec.

Acknowledgements:

"NAT/SLIRP". This universal NAT component is implemented and deployed across a bunch of Open-Source Software programs, including Qemu/KVM, VirtualBox and Virtual Distributed Ethernet (VDE).

It provides source NAT (with PAT), DNS and DHCP services, only lacking the DNS ALG capability and address family translation to become a full-fledged IP-FF babysitter. Conceptually IP-FF babysitter is strongly based on this ideology of integrating NAT+DNS+DHCP services into one.

"NAT64" - "Stateful NAT64: Network Address and Protocol Translation from IPv6 Clients to IPv4 Servers."; [RFC-6146] Basically it provides for AFT (Address Family Translation) "DNS64" - "Domain Name System with IPv6-to-v4 translation.";[RFC-6147] "Network Address Translation - Protocol Translation (NAT-PT)"; [RFC-2766] and criticism of NAT-PT [RFC-4966].

Authors’ Contacts

Alexey Eromenko
Israel

Skype: Fenix_NBK_
EMail: al4321@gmail.com
Facebook: https://www.facebook.com/technologov

INTERNET-DRAFT
Alexey
expiration date: 2017-03-29