464XLAT/NAT64 Optimization

draft-ietf-v6ops-464xlat-optimization-04

Jordi Palet (jordi.palet@theipv6company.com)
Alejandro D’Egidio (adegidio@telecentro.net.ar)
Problem Statement

• In IPv6-only networks using NAT46 (464XLAT, MAP-T), IPv4-only devices flows to dual-stack CDNs/Caches/services are terminated as IPv4, which means extra translations and the subsequent unnecessary overload
  – In many cases this may become a show-stopper

• In equivalent IPv4-only CGN use cases, the CDNs accept “private” addresses (typically 100.64.0.0/10) to avoid exactly the same issues
Typical 464XLAT Deployment

The NAT46/CLAT (WAN side) is connected by IPv6-only to the operator network, which in turn, will have a reverse function, the NAT64 (\[RFC6146\]), known as PLAT (Provider Translator) in the case of 464XLAT. This allows to translate the IPv6-only flow back to IPv4, in order to forward it to Internet.

The translation of the packet headers is done using the IP/ICMP translation algorithm defined in \[RFC7915\] and algorithmically translating the IPv4 addresses to IPv6 addresses following \[RFC6052\].

In the case of 464XLAT, a DNS64 (\[RFC6147\]) optionally is in charge of the synthesis of AAAA records from the A records, so they can use a NAT64, without the need of doing a double-translation by means of the CLAT. However, the DNS64 is not useful for the IPv4-only devices or applications in the LANs, as they will not be able to use the AAAA records.

A typical 464XLAT deployment is depicted in Figure 1.

As it can be observed in the preceding picture, the situation is the same, regardless of in case of a wired network with a CE Router or a cellular network where a UE is connecting other devices (which may be IPv4-only or have IPv4-only apps), by means of a tethering functionality.

If the operator is providing direct access to Content Delivery Networks (CDNs), caches, or other resources, and they are dual-stacked, the situation can be described as shown in Figure 2.

Figure 1: Typical 464XLAT Deployment
IPv6-Capable device

Figure 3: 464XLAT access to CDNs/Caches by IPv6-capable apps
However, if the devices or applications are IPv4-only, for example, most of the SmartTVs and Set-Top-Boxes available today, a non-optimal double translation will occur (NAT46 at the CLAT and NAT64 at the PLAT), as illustrated in Figure 4.

Figure 4: 464XLAT access to CDNs/Caches by IPv4-only apps

Clearly, this is a non-optimal situation, as it means that even if there is a dual-stack service, the NAT46/CLAT translated IPv4 to IPv6 traffic flow, is unnecessarily translated back to IPv4, traversing the stateful NAT64. This has a direct impact in the need to scale the NAT64 beyond what will be actually needed if possible solutions.
IPv4-only device (optimized)

Figure 5: Optimized 464XLAT access to CDNs/Caches by IPv4-only apps
Approach 1: DNS/Routing-based

- CLAT translate A records into AAAA:
  - WKP::A or NSP::A
- CDN/Cache provider configures dedicated interfaces to match WKP::A or NSP::A

```
www.example.com       A    192.0.2.1
CLAT translated to    64:ff9b::192.0.2.1
CDN IPv6 interface must be 64:ff9b::192.0.2.1
Operator must have a specific route to 64:ff9b::192.0.2.1
```

- Issues:
  - Only works if “local/private” connectivity
  - CDN/Cache provider needs to do “something”
Approach 2: CLAT/DNS-proxy-EAMT

• NAT46/CLAT/CE is also a DNS proxy/stub resolver, so an internal interaction can be created.
• This approach uses existing IPv4 and IPv6 addresses (A, AAAA RRs), so no additional complexity for services.

• Steps:
  – Detection of IPv4-only devices
    • Same MAC bound only to IPv4 address, not IPv6.
  – Detection of IPv6-enabled service
  – Creation/maintenance of extended EAMT (RFC7757)
  – Forwarding path for existing EAMT entries via stateful NAT46
Approach 2 Example

- Example

www.example.com  A  192.0.2.1

AAAA  2001:db8::a:b:c:d

EAMT entry  192.0.2.1  2001:db8::a:b:c:d

NAT64/CLAT translated to  2001:db8::a:b:c:d

CDN IPv6 interface already is  2001:db8::a:b:c:d

Operator already has specific route to  2001:db8::a:b:c:d

1. A query for www.example.net A RR is received
2. www.example.net  A  192.0.2.1
3. www.example.net  AAAA  2001:db8::e:e:f:f
4. A conflict has been detected
5. The existing EAMT entry for 192.0.2.1 is set to stale
   (it can be used to continue existing previous connections, but not new ones)
Approach 2: Additional Considerations

- Behavior in case of multiple A/AAAA RRs
- Behavior in case of presence/absence of DNS64
- Behavior when using literal addresses or non IPv6-APIs
- Behavior in case of Foreign DNS
  - Devices/apps using other DNS
  - DNS privacy/encryption
  - DNS modified by user in OS
  - DNS modified by user in CE
  - Combinations of above
- False detection of a dual-stack host as IPv4-only
- Behavior in presence of HE
- Troubleshooting implications
Approach 3: CLAT-provider-EAMT

- Similar to previous one, but no "automated" EAMT
- Operator must push or CE must pull the table
- It will work even if user change DNS for STB, SmartTV, …
- More control from the operator
  - EAMT pairs may be built “apart” from DNS
- Issues:
  - Increase complexity
    - Is the benefit worth for it?
  - The CDN/cache provider needs to provide API to update the EAMT, or the operator use their own caches to build it
Questions?

• Optimization disabled by default?
  – If there is no 464XLAT being used is not enabled.
  – If there is no NAT64 discovered is not enabled.

• Clients ignoring the TTL
  – TTL is being used by the EAMT

• Consumer electronics using alternate resolvers
  – Supported

• Security considerations
  – We still don’t see any additional security issue created by this approach