464XLAT: Breaking Free of IPv4

Cameron.Byrne@T-Mobile.com

APRICOT 2014

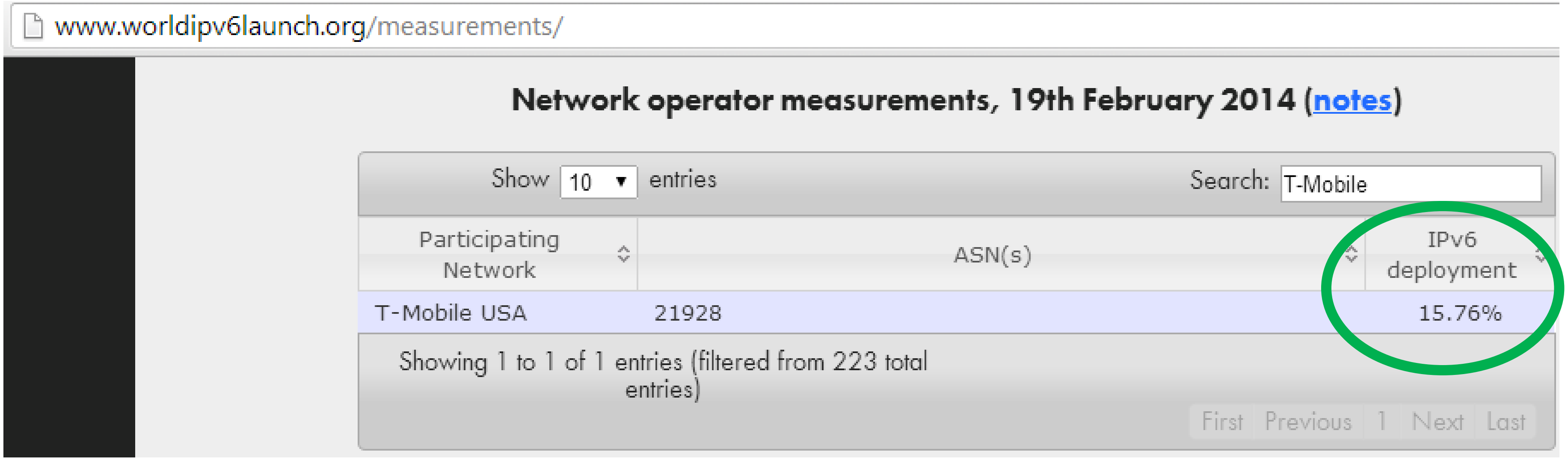
Background

- T-Mobile US is a GSM / UMTS / LTE provider in the USA with 45+ Million subscribers
- In 2008, T-Mobile launched the first Android phone. This dramatically changed the mobile data dynamics – more devices, connected for a longer time, all needing IP addresses
- T-Mobile embraced the concept of IPv6-only, since dual-stack required IPv4 that was not available
- NAT64 / DNS64 was a good solution that did not require IPv4 on each client, but some applications failed to work on IPv6-only networks. It is not acceptable to break Skype or Netflix, applications that require IPv4
- T-Mobile, in partnership with NEC and JPIX, documented 464XLAT in the IETF as RFC6877 to overcome the limitations of NAT64 by adding a NAT46 into the client (CLAT)
- Android 4.3 introduced support for 464XLAT in October 2013

Results Are Important

- T-Mobile US launched 5 Android phones with 464XLAT as the default in the last 5 months, all Android 4.3+ phones will be 464XLAT in the future at T-Mobile US
- 3.6 million unique IPv6 subscribers in the first 5 months are active on the network
- http://www.worldipv6launch.org/measurements/ measurements show 15.76% of all T-Mobile connections are now IPv6, as of February 21, 2014
- Over 50% of IPv6-user traffic is end-to-end IPv6 (no translation needed) ← ***This saves money and makes the network simpler***

15.76 of T-Mobile US Connections use 464XLAT



www.worldipv6launch.org/measurements/

Network operator measurements, 19th February 2014 (notes)

Show 10 entries Search: T-Mobile

Participating Network	ASN(s)	IPv6 deployment
T-Mobile USA	21928	15.76%

Showing 1 to 1 of 1 entries (filtered from 223 total entries)

Default 464XLAT Phones at T-Mobile US

Samsung
Note 3

Google / LG
Nexus 5

Samsung
Mega

464XLAT Phones

Sony Z1s

Galaxy Light

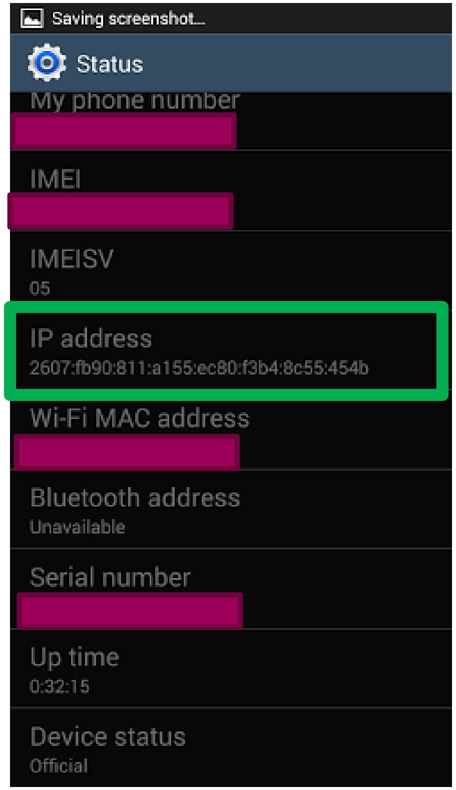
464XLAT allows for full functionality on IPv6-only networks

- Dual-stack does not solve the IPv4 number scarcity issue
- IPv6-only + NAT64/DNS64 is very good, but not good enough for full IPv4 replacement (web and email work, but Skype does not work)
- IPv6-only + 464XLAT
  - Solves IPv4 numbering issue by not assigning IPv4 to clients
  - Decouples edge growth from IPv4 availability
  - IPv4-only applications like Skype work on an IPv6-only network because 464XLAT translates IPv4 on the phone to IPv6 on the network

IPv6 deployment is ~~easy~~ achievable

- T-Mobile USA did not spend any CapEx on IPv6
- Only introduce 464XLAT on new phones, so we do not disrupt any existing services
- Innovative thinking helps reduce deployment costs (hash 128 bit numbers into 32 bit fields in billing records)
- IPv6 will save money in your network (less NAT/CGN, no need to buy IPv4 addresses, …)

In fact, with roaming, we can show Chunghwa in Taiwan and MY MAXIS in Malaysia support IPv6 today in the Radio Access Network (RAN)


Saving screenshot...

Status

My phone number
[redacted]

IMEI
[redacted]

IMEISV
05

IP address
2607:fb90:811:a155:ec80:f3b4:8c55:454b

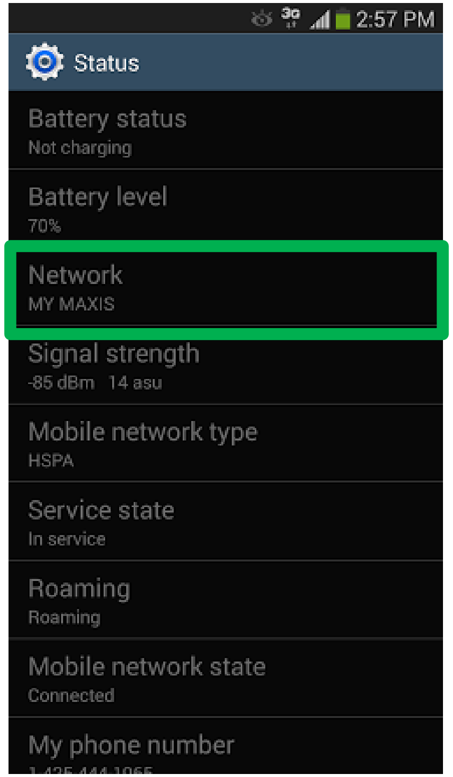
Wi-Fi MAC address
[redacted]

Bluetooth address
Unavailable

Serial number
[redacted]

Up time
0:32:15

Device status
Official



Status

Battery status
Not charging

Battery level
70%

Network
MY MAXIS

Signal strength
-85 dBm 14 asu

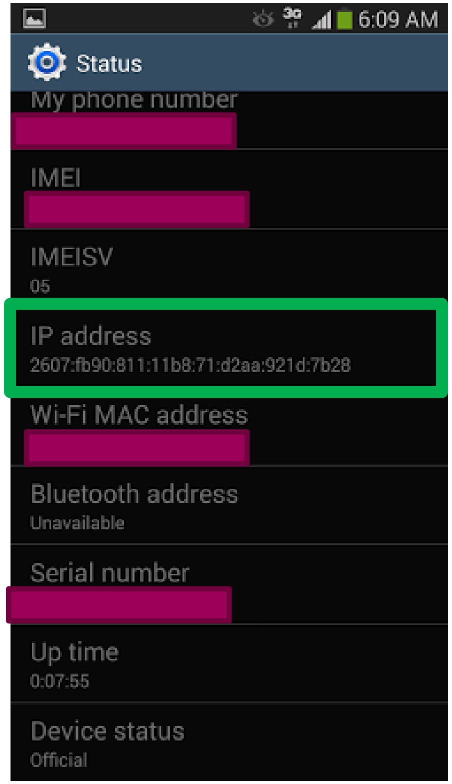
Mobile network type
HSPA

Service state
In service

Roaming
Roaming

Mobile network state
Connected

My phone number
1 425 444 1065



Status

My phone number
[redacted]

IMEI
[redacted]

IMEISV
05

IP address
2607:fb90:811:11b8:71:d2aa:921d:7b28

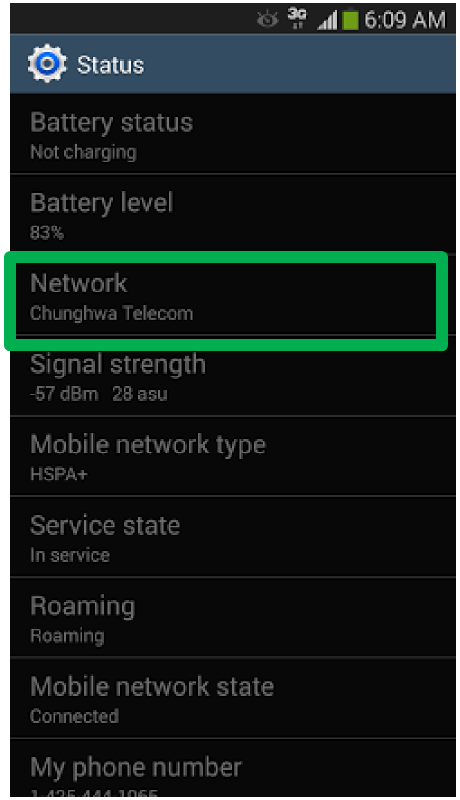
Wi-Fi MAC address
[redacted]

Bluetooth address
Unavailable

Serial number
[redacted]

Up time
0:07:55

Device status
Official



Status

Battery status
Not charging

Battery level
83%

Network
Chunghwa Telecom

Signal strength
-57 dBm 28 asu

Mobile network type
HSPA+

Service state
In service

Roaming
Roaming

Mobile network state
Connected

My phone number
1 425 444 1065

Which Platforms Supports 464XLAT Today?

YES	NO
Android 4.3+	Blackberry
	Apple
	Windows Phone (?)

IMPORTANT!

- Anything that is natively IPv6 enabled does not require any sort of translation, 464XLAT is idle and transparent for any IPv6 end-to-end flow
- IPv6 end-to-end just works!
- 464XLAT is only for service and applications that are using LEGACY IPV4
- As more and more services transition to IPv6, 464XLAT is engaged less and less
- 464XLAT is an IPv4 EXIT STRATEGY

THE TECHNICAL DETAILS

464XLAT is just a set of building blocks

- Stateless NAT64 (RFC6145)
  - Client side translation CLAT
- Statefull NAT64 (RFC6146)
  - Provider site translation PLAT
- DNS64 (RFC 6147)
  - When the FQDN does not have a AAAA record, DNS64 dynamically creates one that allows the client to use IPv6 and the network translates from IPv6 to IPv4 at the NAT64
- Prefix64 Discovery (RFC 7050)
  - Queries for the well-known FQDN ipv4only.arpa, which is by definition IPv4-only. If there is a AAAA response provided, then it is known that a DNS64 is in the path

3 Scenarios in 464XLAT

1. End-to-end IPv6: Facebook, Google, Wikipedia, Yahoo, Youtube … IPv6->IPv6
2. Application supports IPv6 (web browser) but the server is only IPv4 (www.amazon.com, www.myspace.com, …), so DNS64/NAT64 translates IPv6->IPv4
3. Application does not support IPv6 (Skype, Whatsapp, …), the client must provide a stateless NAT46 to the application and stateful NAT64 must be in the network: IPv4->IPv6->IPv4

How does Stateless NAT64 work?

- Algorithmically map IPv4 addresses to IPv6 addresses, bidirectional, 1 to 1
  - Not dynamic
  - Deterministic
  - Maps all of IPv4's 32 bits into an IPv6 /96 (or larger prefix)
- Defined in RFC6145
- Example
  - 2001:db8::10.1.1.1 <-> 10.1.1.1
  - 2001:db8::10.2.2.2 <-> 10.2.2.2
  - 2001:db8::www.example.com <-> ipv4 www.example.com

How does Stateful NAT64 work?

- Dynamically translate IPv6 packets to IPv4 packets
  - Dynamic
  - Not deterministic (translation based on available IPv4 pool)
  - Translation state is short-lived and based on session creation and termination
- Defined in RFC6146
- Example
  - Before translation
    - TCP source 2001:db8:abcd::ffff port 555 # client address
    - TCP destination 2001:db8:1234::10.1.1.1 port 80 # NAT64 address
  - After translation
    - TCP source 192.168.1.1 port 555 # 192.168.1.1 available from NAT64 pool
    - TCP destination 10.1.1.1 port 80 # Last 32 bits of IPv6 destination

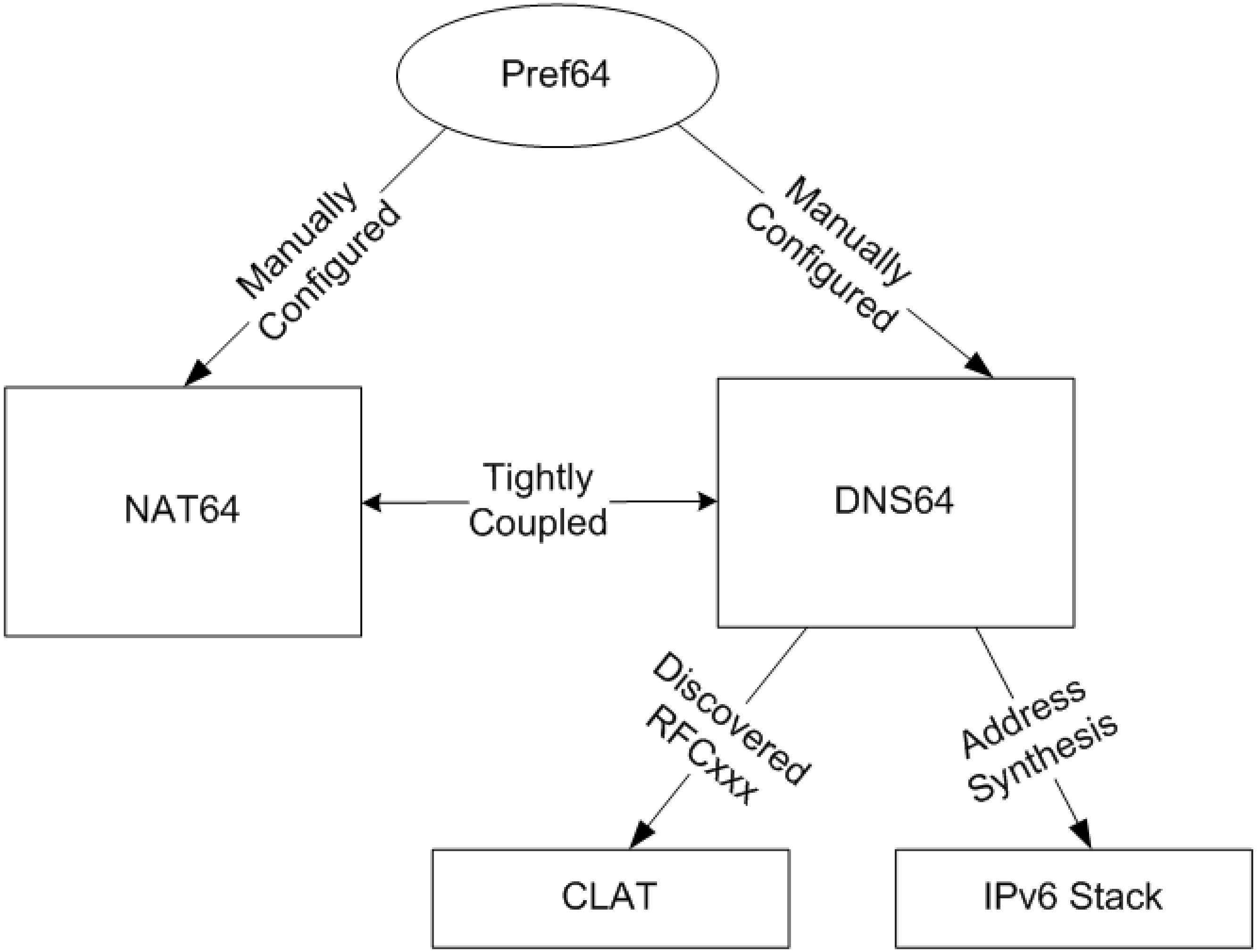
How does DNS64 work?

- When an FQDN does not have a AAAA record, the DNS64 will synthetically create one based on a network defined Pref64
- The pref64 is a prefix hosted on the NAT64 for translation
- Example without DNS64
  - Query = a and aaaa for www.example.com
  - Answer = a = 10.1.1.1, aaaa = NO ERROR
- Example with DNS64
  - Query = a and aaaa for www.example.com
  - Answer = a = 10.1.1.1 AND aaaa = 2001:db8::10.1.1.1

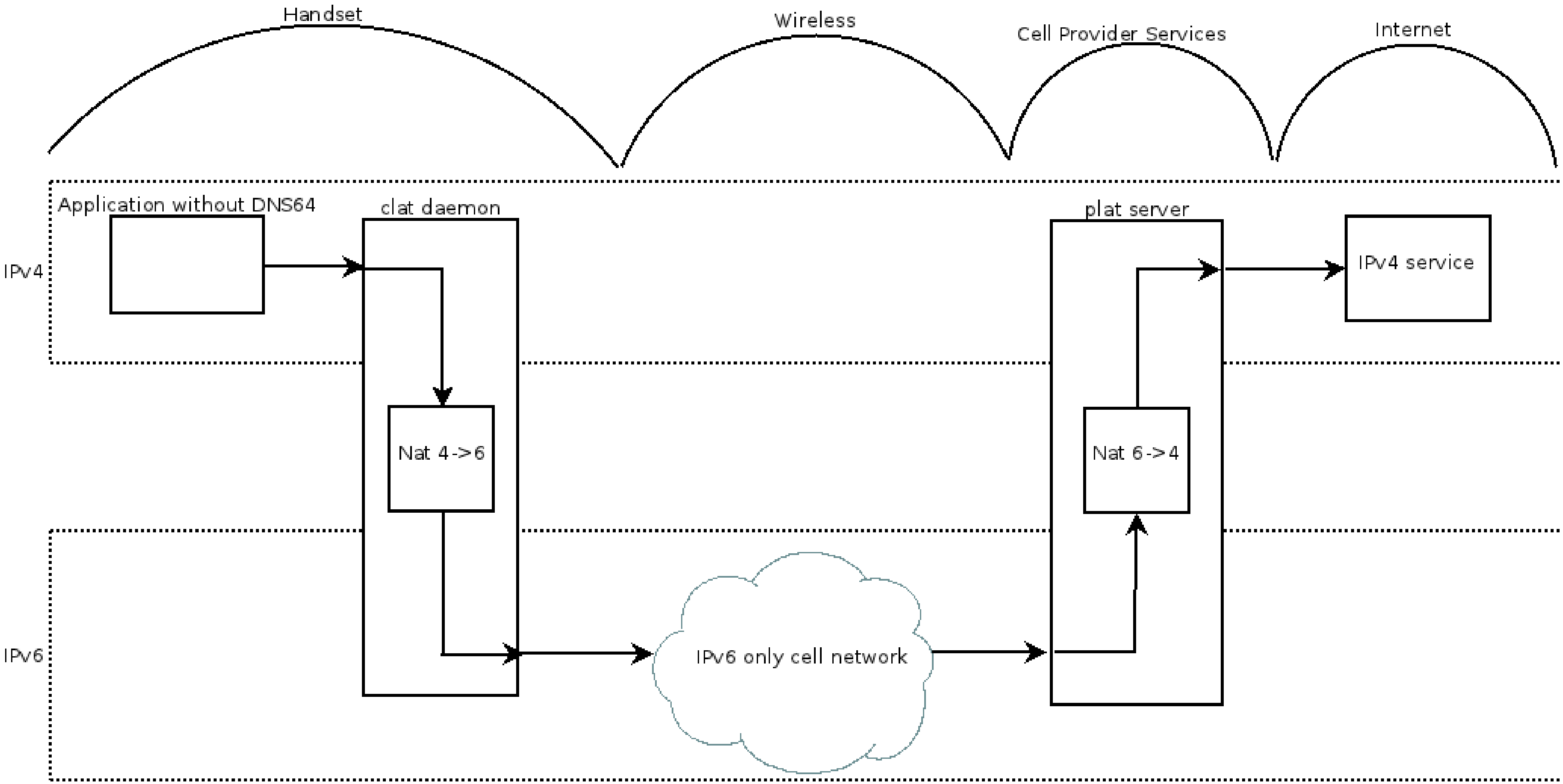
How is the Pref64 discovered on the client?

- Pref64 is topologically located on the NAT64
- The DNS64 forces clients to send traffic to the NAT64 for translation from IPv6 to IPv4
- Automatic discovery of Pref64 is defined in RFC 7050
- The client will lookup the well-known FQDN ipv4only.arpa. If a AAAA record is presented for this well-know IPv4-only FQDN, the clientcan parse the response to find the Pref64 used within this network

Pref64 Configuration Information Flow



How to make EVERYTHING work on IPv6-only?

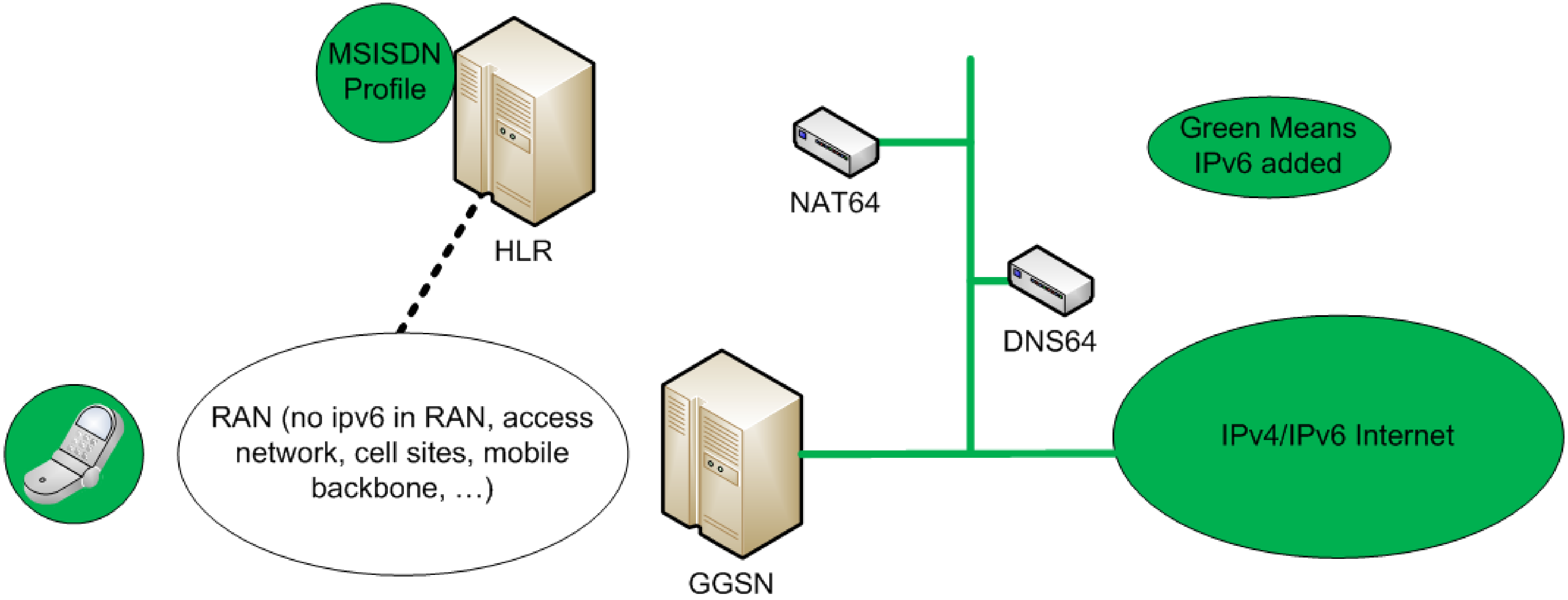


T · ·Mobile· stick together®

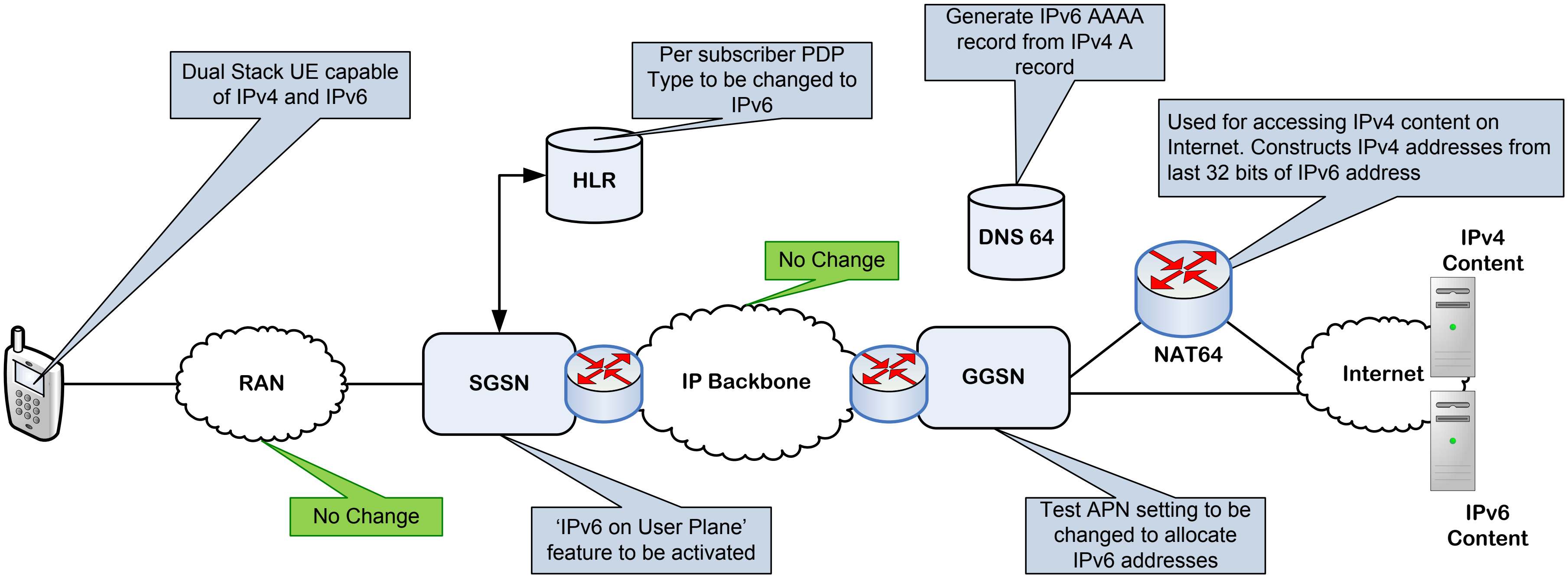
Zoom Out: What does this look like in the context of 3GPP GSM / UMTS / LTE ?

High Level View of IPv6 deployment:

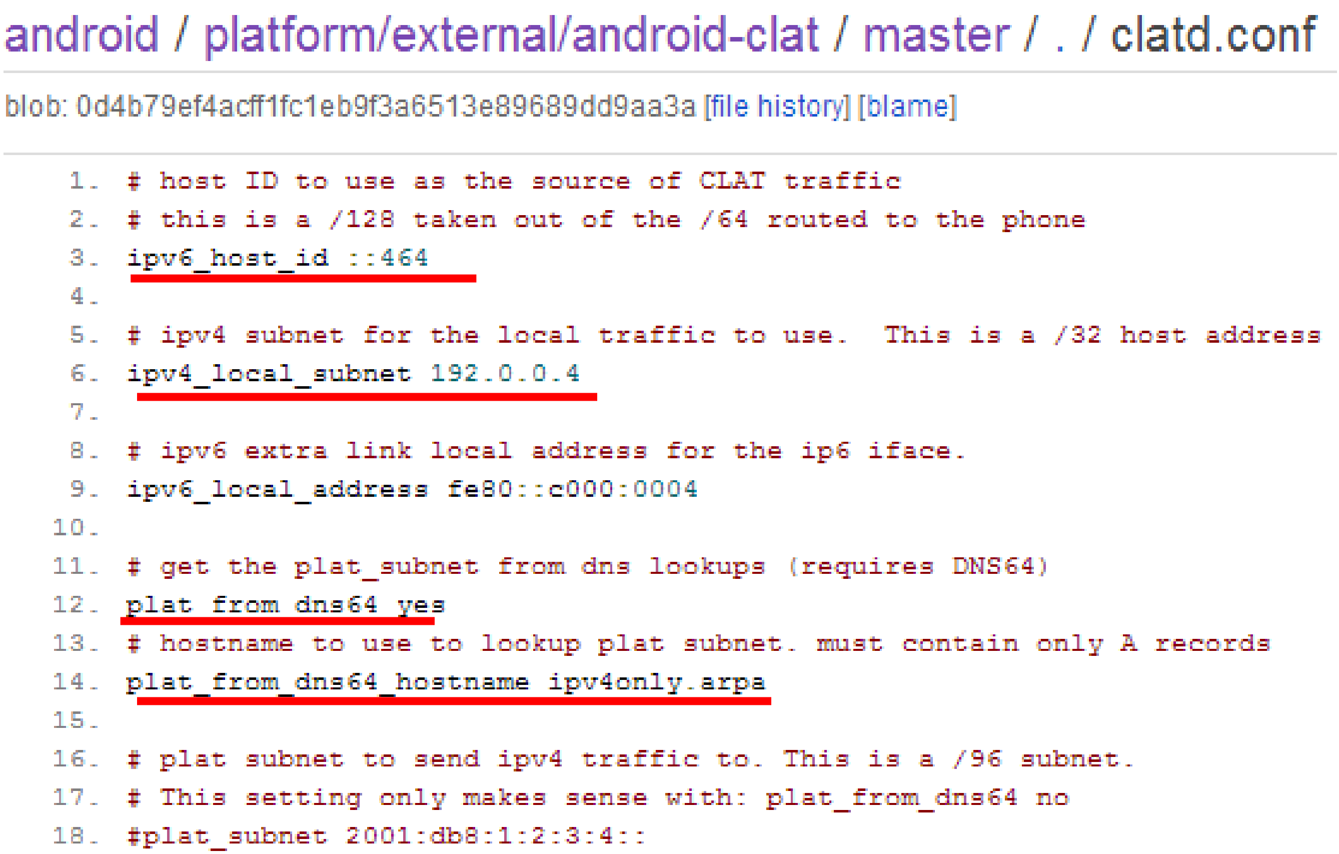
Phone, HLR profile, GGSN, NAT64, IPv6 ISP



Impact to Network Entities


Dual Stack UE capable of IPv4 and IPv6
Per subscriber PDP Type to be changed to IPv6
Generate IPv6 AAAA record from IPv4 A record
Used for accessing IPv4 content on Internet. Constructs IPv4 addresses from last 32 bits of IPv6 address
HLR
DNS 64
No Change
IPv4 Content
NAT64
RAN
SGSN
IP Backbone
GGSN
Internet
No Change
'IPv6 on User Plane' feature to be activated
Test APN setting to be changed to allocate IPv6 addresses
IPv6 Content

Zoom in: What does the default Android configuration look like?: clatd.conf

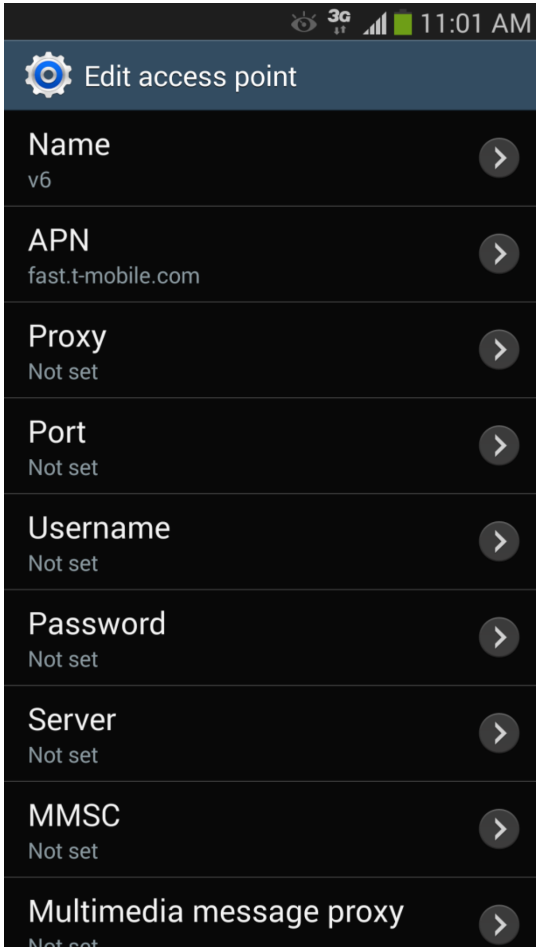


android / platform/external/android-clat / master / . / clatd.conf

blob: 0d4b79ef4acff1fc1eb9f3a6513e89689dd9aa3a [file history] [blame]

```
# host ID to use as the source of CLAT traffic
# this is a /128 taken out of the /64 routed to the phone
ipv6_host_id ::464

# ipv4 subnet for the local traffic to use.  This is a /32 host address
ipv4_local_subnet 192.0.0.4

# ipv6 extra link local address for the ip6 iface.
ipv6_local_address fe80::c000:0004

# get the plat_subnet from dns lookups (requires DNS64)
plat_from_dns64 yes
# hostname to use to lookup plat subnet. must contain only A records
plat_from_dns64_hostname ipv4only.arpa

# plat subnet to send ipv4 traffic to. This is a /96 subnet.
# This setting only makes sense with: plat_from_dns64 no
#plat_subnet 2001:db8:1:2:3:4::
```

Zoom in: What does the phone configuration look like: APN Settings



Edit access point

Name
v6

APN
fast.t-mobile.com

Proxy
Not set

Port
Not set

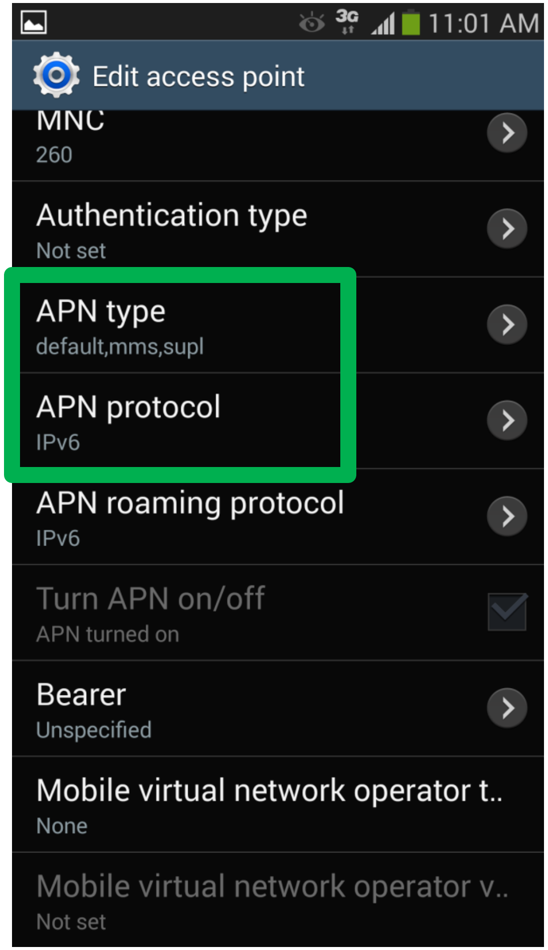
Username
Not set

Password
Not set

Server
Not set

MMSC
Not set

Multimedia message proxy



Edit access point

MNC
260

Authentication type
Not set

APN type
default,mms,supl

APN protocol
IPv6

APN roaming protocol
IPv6

Turn APN on/off ☑
APN turned on

Bearer
Unspecified

Mobile virtual network operator t..
None

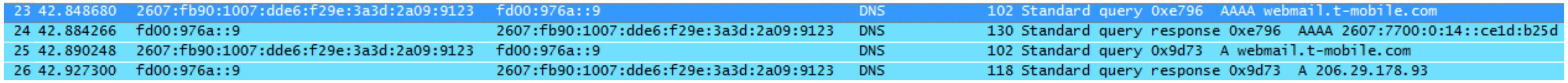
Mobile virtual network operator v..
Not set

In Android 4.3, “APN Protocol IPv6” for the “APN Type default” triggers the use of 464XLAT by default

IPv6 = 464XLAT

TIME FOR WIRESHARK

Like most things, we start with DNS

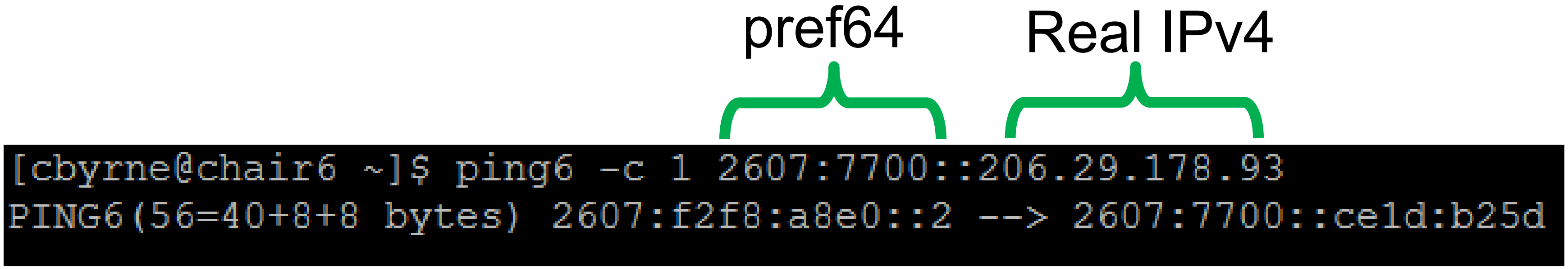


```
23 42.848680  2607:fb90:1007:dde6:f29e:3a3d:2a09:9123  fd00:976a::9                             DNS         102 Standard query 0xe796  AAAA webmail.t-mobile.com
24 42.884266  fd00:976a::9                             2607:fb90:1007:dde6:f29e:3a3d:2a09:9123  DNS         130 Standard query response 0xe796  AAAA 2607:7700:0:14::ce1d:b25d
25 42.890248  2607:fb90:1007:dde6:f29e:3a3d:2a09:9123  fd00:976a::9                             DNS         102 Standard query 0x9d73  A webmail.t-mobile.com
26 42.927300  fd00:976a::9                             2607:fb90:1007:dde6:f29e:3a3d:2a09:9123  DNS         118 Standard query response 0x9d73  A 206.29.178.93
```

- The client is IPv6-only towards the network, but the host OS thinks it is dual-stack since it has an IPv4 CLAT interface and a native IPv6 radio interface
- So, the client does a query for DNS “A” and “AAAA” records
- The DNS64 responds with a synthesized AAAA and the real A
- The synthesized AAAA = Pref64 + real IPv4

Quick Check

- Does the synthesized AAAA match the pref64 + real A?



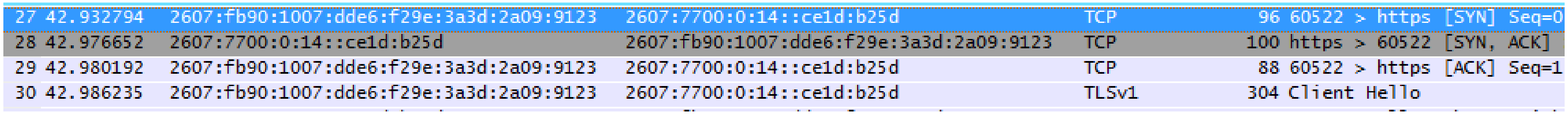
pref64

Real IPv4

```
[cbyrne@chair6 ~]$ ping6 -c 1 2607:7700::206.29.178.93
PING6(56=40+8+8 bytes) 2607:f2f8:a8e0::2 --> 2607:7700::ce1d:b25d
```

T-Mobile stick together®

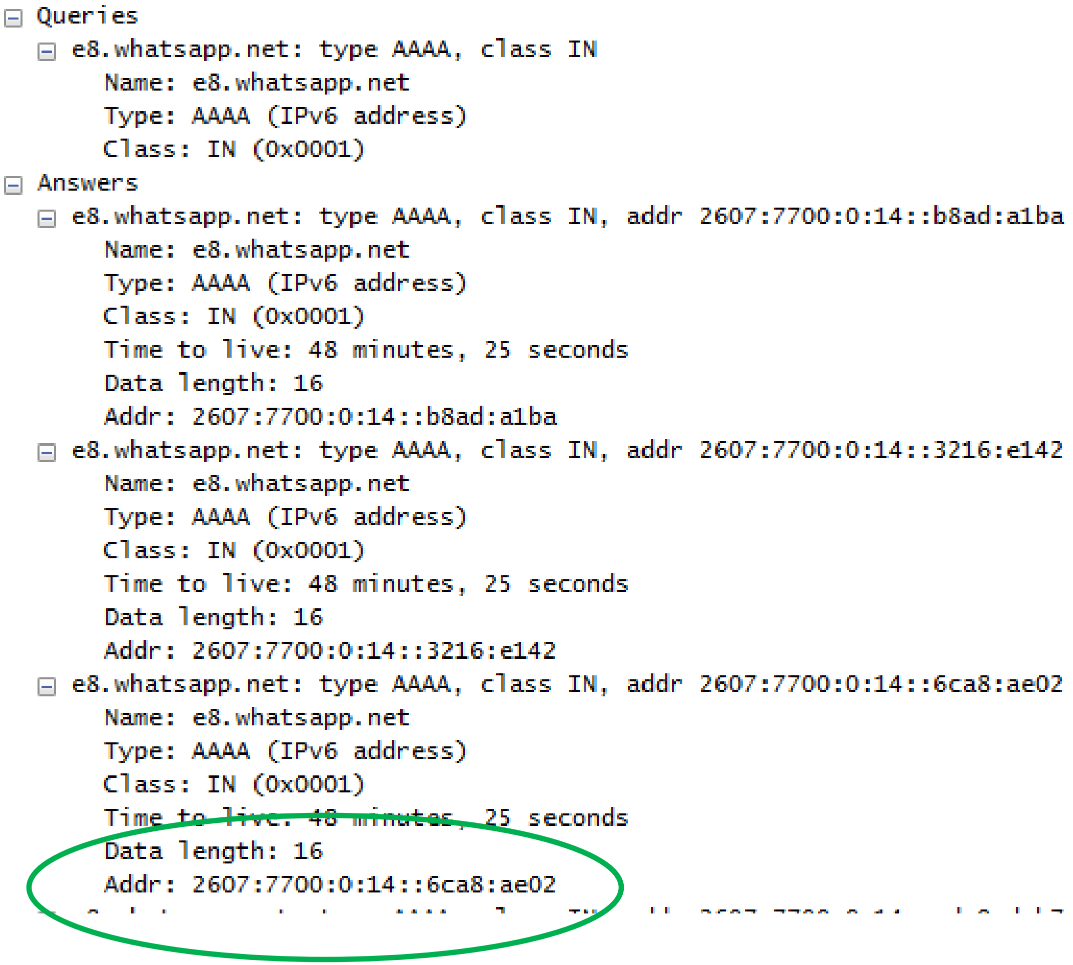
Next, the UE selects the IPv6 DNS response, and starts TCP



```
27 42.932794   2607:fb90:1007:dde6:f29e:3a3d:2a09:9123   2607:7700:0:14::ce1d:b25d                 TCP       96 60522 > https [SYN] Seq=0
28 42.976652   2607:7700:0:14::ce1d:b25d                 2607:fb90:1007:dde6:f29e:3a3d:2a09:9123   TCP      100 https > 60522 [SYN, ACK]
29 42.980192   2607:fb90:1007:dde6:f29e:3a3d:2a09:9123   2607:7700:0:14::ce1d:b25d                 TCP       88 60522 > https [ACK] Seq=1
30 42.986235   2607:fb90:1007:dde6:f29e:3a3d:2a09:9123   2607:7700:0:14::ce1d:b25d                 TLSv1    304 Client Hello
```

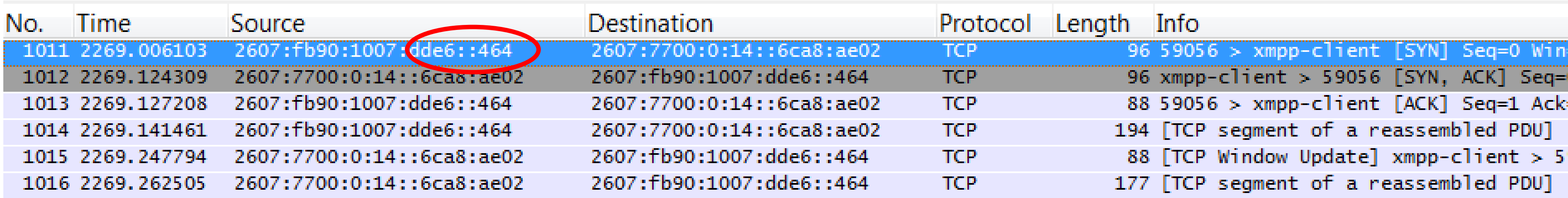
- From the client perspective, this is a native IPv6 end-to-end flow
- But, we know that the DNS is a synthesized AAAA and the client is actually sending its packets to the NAT64 for IPv6->IPv4 stateful translation
- This is just DNS64 / NAT64, no client-side translation needed for this scenario

The full case of 464XLAT double translation: WhatsApp



```
⊟ Queries
  ⊟ e8.whatsapp.net: type AAAA, class IN
      Name: e8.whatsapp.net
      Type: AAAA (IPv6 address)
      Class: IN (0x0001)
⊟ Answers
  ⊟ e8.whatsapp.net: type AAAA, class IN, addr 2607:7700:0:14::b8ad:a1ba
      Name: e8.whatsapp.net
      Type: AAAA (IPv6 address)
      Class: IN (0x0001)
      Time to live: 48 minutes, 25 seconds
      Data length: 16
      Addr: 2607:7700:0:14::b8ad:a1ba
  ⊟ e8.whatsapp.net: type AAAA, class IN, addr 2607:7700:0:14::3216:e142
      Name: e8.whatsapp.net
      Type: AAAA (IPv6 address)
      Class: IN (0x0001)
      Time to live: 48 minutes, 25 seconds
      Data length: 16
      Addr: 2607:7700:0:14::3216:e142
  ⊟ e8.whatsapp.net: type AAAA, class IN, addr 2607:7700:0:14::6ca8:ae02
      Name: e8.whatsapp.net
      Type: AAAA (IPv6 address)
      Class: IN (0x0001)
      Time to live: 48 minutes, 25 seconds
      Data length: 16
      Addr: 2607:7700:0:14::6ca8:ae02
```

SYN is sent from the CLAT address



No.	Time	Source	Destination	Protocol	Length	Info
1011	2269.006103	2607:fb90:1007:dde6::464	2607:7700:0:14::6ca8:ae02	TCP	96	59056 > xmpp-client [SYN] Seq=0 Win
1012	2269.124309	2607:7700:0:14::6ca8:ae02	2607:fb90:1007:dde6::464	TCP	96	xmpp-client > 59056 [SYN, ACK] Seq=
1013	2269.127208	2607:fb90:1007:dde6::464	2607:7700:0:14::6ca8:ae02	TCP	88	59056 > xmpp-client [ACK] Seq=1 Ack
1014	2269.141461	2607:fb90:1007:dde6::464	2607:7700:0:14::6ca8:ae02	TCP	194	[TCP segment of a reassembled PDU]
1015	2269.247794	2607:7700:0:14::6ca8:ae02	2607:fb90:1007:dde6::464	TCP	88	[TCP Window Update] xmpp-client > 5
1016	2269.262505	2607:7700:0:14::6ca8:ae02	2607:fb90:1007:dde6::464	TCP	177	[TCP segment of a reassembled PDU]

Remember, we set the clatd.conf to use the IID of ::464 for CLAT translations

The MMS situation

- The Android MMS function communicates directly to the modem and by-passes the normal OS networking stack
- Frequently MMS is its own APN
- This means 464XLAT is bypassed, 464XLAT only works on the default APN, not special APNs like SUPL and MMS
- Solutions
  - Use an FQDN, DNS64 still works fine
  - If you cannot use an FQDN, manually use a NAT64 literal instead of the IPv4 literal (pref64 + ipv4 literal)

Security: Follow the rule of least privilege

- Filter access to the DNS sever
- Filter access to the Pref64 on the NAT64
- Using ULA Pref64 will NOT work well since Android prefers IPv4 over IPv6 ULA. This results in 100% CLAT translations for IPv4 resources

Summary

- IPv4 does not fit the business needs to grow the edge of our networks fueled by growth from internet of things and cloud
- IPv6 works today and is deployed on some of the largest edge networks
- 464XLAT allows networks to grow without many public IPv4 addresses
- IPv6 deployment in 3GPP GSM / UMTS /LTE is achievable today

Big Picture: We must avoid the Internet’s largest growth engine (mobile) from being indefinitely tied to scarce IPv4 and fragile stateful NAT44.