

The IPv6-Only Network

Merit IPv6 Transition Readiness Program

February 2018

Ann Arbor, MI



A YouTube video of this presentation is linked to:

<http://ipv6hawaii.org/?p=388>

"The IPv6 Only Network"



People Are Interested In This

- My original canvas of network trainers, last April to find out if anybody had a “NAT64 training” got a unanimous, “no, but we’ve been interested in doing that”
- Several people have volunteered plans for campus deployments, either as a try-and-see, or to understand it as a tool
- IETF, LACNIC, APNIC (at least) have had it at their meetings.
- WIDE has done “camps”, sort of LAN parties for NAT64 enthusiasts, with strong reporting and results tracking



Definition of “IPv6-only Network”

- A network where the end-user-devices, have only IPv6 addresses.
 - More importantly, their local gateway routes only IPv6, no IPv4
- Possibly also where most routers and infrastructure have only IPv6 addresses
- IPv4 is offered to users as a service, over IPv6.



Next Friday, Close Of Business

- Pretend we're doing a general purpose end-user-device WiFi network for 500 devices, to be working next Friday, 5 PM.
- Rules of today's game
 - We're talking about solutions that are available to us to deploy in the space of the next week.
 - Perhaps assuming vendor equipment donations
 - Perhaps assuming open source Linux-based routers and stuff
 - Positive user experience is key
 - Elegance is Priority #12,397
 - The “morally-right” way to assign addresses or DNS servers or etc is a distant conversation
 - We admit that breaking DNSSEC is something we'd rather not do, but also that nobody would notice, when we deploy, next week
 - Interesting-new-internet-drafts, or even RFCs are irrelevant, unless they already have production-ready, supported implementations

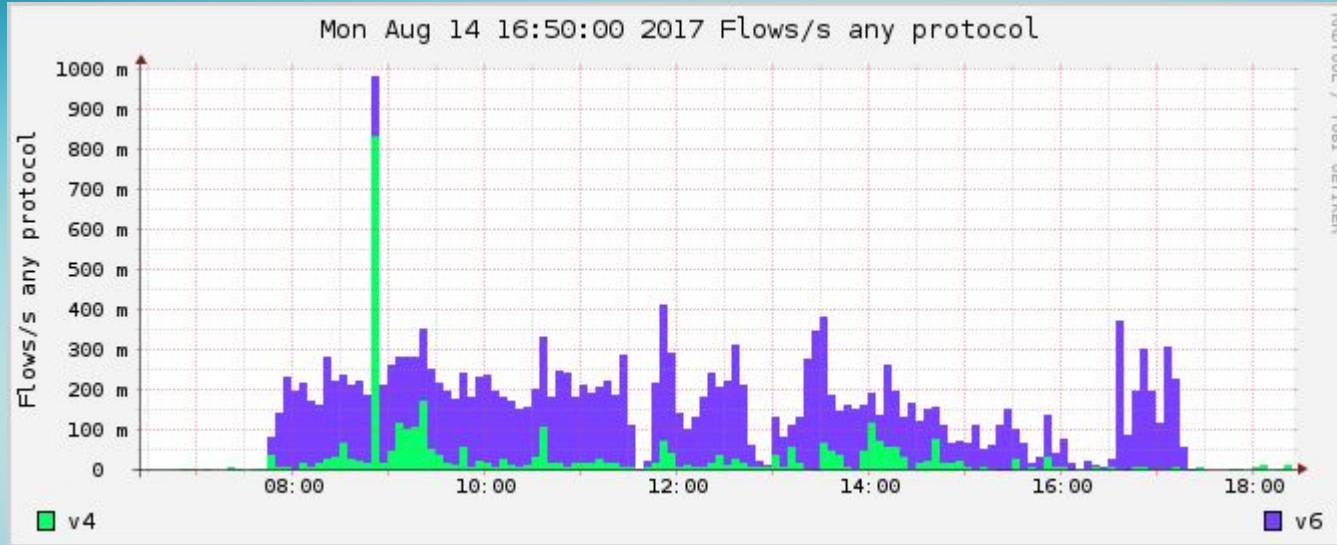


Experiences - U. Hawaii

- At U. Hawaii ITS, we've been dabbling in the art of NAT64/DNS64 since July 2017.
- Several of us have done our work through such a setup on various days, noting some issues, but by and large getting work done without serious problems.
- In August, we hosted an OIN Workshop with about 40 people attending. We offered a DNS64/NAT64 SSID during that meeting, and many people just used it all day, without problems.
- I've been using IPv6-only with IPv4AAS as [DNS64/NAT64/464XLAT] in my office for about 2 months. It is indistinguishable from our normal dual stack network.



OIN Workshop ipv6-only SSID



V6 flows are native-only

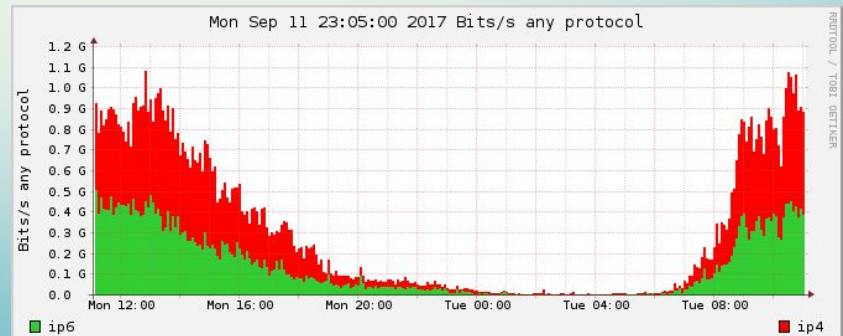
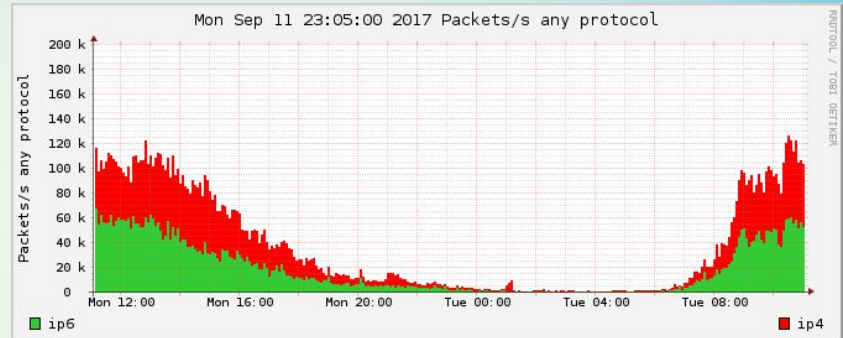
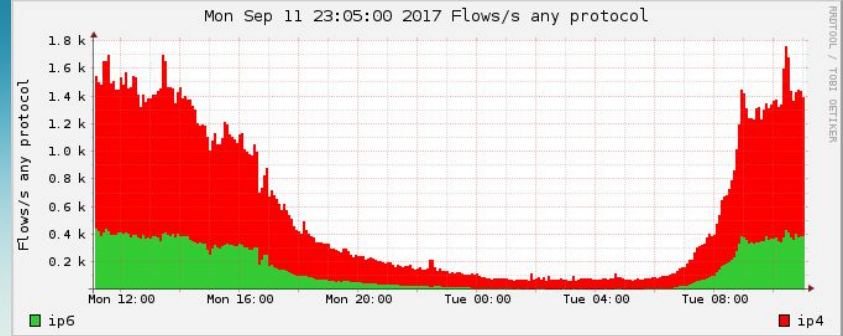


U. Hawaii Manoa Wireless (dual stack)

Flows: v6 ~ 30%

Packets: v6 ~ 60%

Bits: v6 ~ 50%



The Things You Have At Home

- Had a Cisco ASR1001, not-in-service, with NAT64-capable software
 - Had to upgrade IOS-XE on a Cisco 4500 to get RDNSS on user segment
 - Built VMs on ProxMox NAT64, 464CLAT, DNS64, DHCP6, DHCP4
 - Put a “ipv6-only” SSID on the IT building Aruba WiFi
-
- Could have done DNS64/NAT64 with the ASR1001 and Google’s DNS64
 - Or with Jool NAT64 and Google’s DNS64
 - Or you can just use the live demos at Go6 Lab in Slovenia
 - <https://go6lab.si/current-ipv6-tests/nat64dns64-public-test/>



Experiences - 3 Perspectives

- Equipment Vendor
 - Customer drivers
 - Adversity to change
 - Strategic mandates
 - San Jose Campus Building 23
- Network admin (self)
 - Simplicity and automation
 - Performance
 - Embrace the future - self-initiatives, self-motivated
- Network admins (other)
 - Adversity to change
 - Jokes (doomsday)
 - Fear to learn?
 - Too complex



The Value Of IPv6 Only

- Operational simplicity
 - Single stack support.
- Enhanced security
 - Reduction of attack surface to a single IP stack
- Limits deployment of “real” IPv4 addresses, so that they may be used efficiently
 - IPv4 addresses can only become more costly until IPv6 is the norm
- Provides direct, non-NAT connectivity to IPv6 connected resources
 - Which is a larger proportion of your traffic than you might think
- Your commitment to IPv4 deployment gets shallower, rather than deeper.
- Dual Stack **does not** reduce reliance on IPv4 addresses
- Using IPv6-only connected clients **does** reduce reliance on IPv4 addresses



The Value Of IPv6 Only

- North American IPv6 Summit
 - Shift in mindset
 - Examples: LinkedIn, Cisco building 23, T-mobile



To DHCPv6-only or DHCPv6-SLAAC?

- “Religious” battle
 - Who is right?
 - What does being “right” have to do with anything? (Betamax was better)
- DHCPv6-only
 - Android anyone?
- SLAAC (RDNSS)
 - RA Options
- DHCPv6-SLAAC (RDNSS)
 - Catch all

```
ipv6 dhcp pool MAINv6
```

```
dns-server 2620:0:CCC::2
```

```
dns-server 2620:0:CCD::2
```



IPv4-As-A-Service

- More of a philosophical construct than a technical one
- Refers to providing IPv4 content access to IPv6-only hosts
- Generally treated as a futuristic concept
- When should we declare that it's time for IPv4 As A Service?
 - How about now?
- At least one vendor
 - <https://www.retevia.net/>



Special diagnostic sites

- v4o.tut.uhnet.net -
 - v4o has only an IPv4 address
- v6o.tut.uhnet.net -
 - v6o has only an IPv6 address

Both are using public addresses

Sometimes, you just need to be sure you're getting v4 only or v6 only content.

Browser plugins:

- Chrome, Firefox: IPvFoo
(<https://chrome.google.com/webstore/detail/ipvfoo/ecanpcehffngcegjmadlcijfolapggal?hl=en>)
- Firefox: IPvFox (<https://addons.mozilla.org/en-US/firefox/addon/ipvfox/>)



V4-only diag page: The best thing I ever did



IPv4 only web page



Server address: [redacted]#80
Client address: [redacted]#45359

Your browser: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/61.0.31


IPv4 only web page

Server address: [redacted]#80
Client address: [redacted]#45358

Your browser: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/61.0.3163.100 Safari/537.36

<p>image by v4 literal</p>  <p>By Abhijit Tembhekar from Mumbai, India (Nikon D80 Apple) [CC BY 2.0 (http://creativecommons.org/licenses/by/2.0), via Wikimedia Commons</p>	<p>image by name</p>  <p>By Evan-Amos (Own work) [CC BY-SA 3.0 (https://creativecommons.org/licenses/by-sa/3.0), via Wikimedia Commons</p>
---	---

Working NAT64, Working 464XLAT

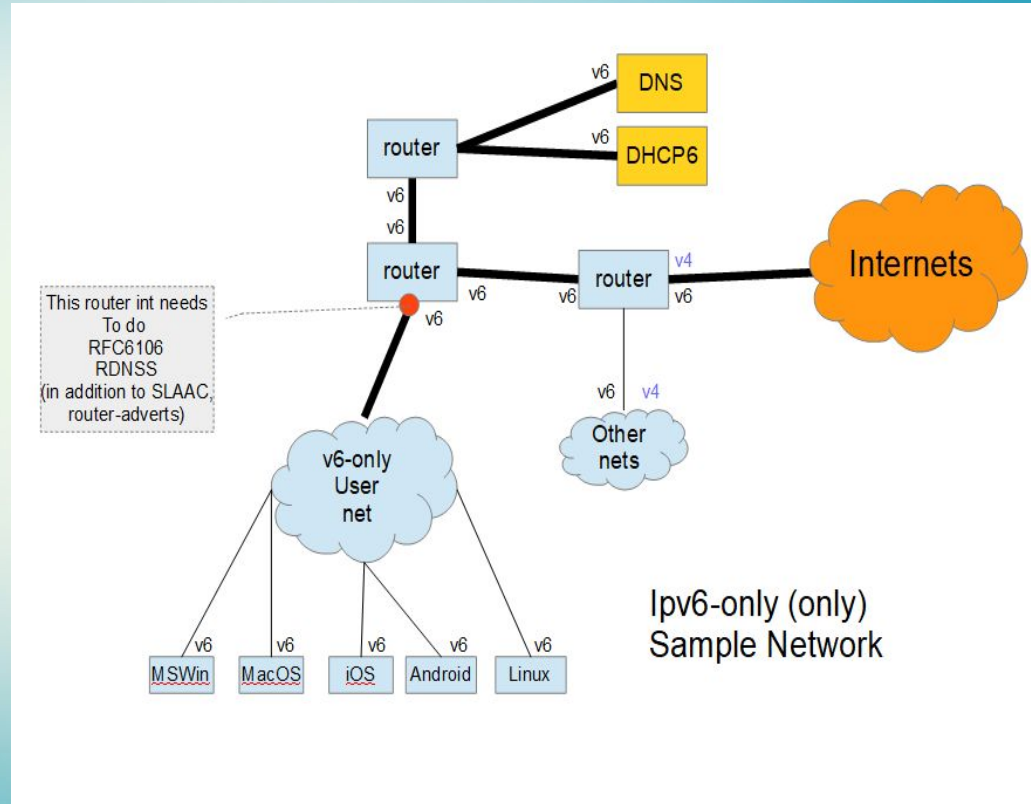
<p>image by v4 literal</p> <p>By Abhijit Tembhekar from Mumbai, India (Nikon D80 Apple) [CC BY 2.0 (http://creativecommons.org/licenses/by/2.0), via Wikimedia Commons</p>	<p>image by name</p>  <p>By Evan-Amos (Own work) [CC BY-SA 3.0 (https://creativecommons.org/licenses/by-sa/3.0), via Wikimedia Commons</p>
--	---

Working NAT64, Non-working 464CLAT



Scenario 1: The IPv6-only (only) Network

- Offers IPv6 connectivity
- No IPv4AAS
- Offers DHCP6 and RA-RDNSS to supply DNS server
- Offers SLAAC addresses because Android doesn't do DHCP6
- Content is sparse, but surprising, if you know where to look.



The IPv6-only (only) Reachability

How well can a user surf the Net without using *any* IPv4?

(Accessing only resources that are fully available over IPv6)

Working:

CNN.com*, Netflix.com*, YouTube.com*, FoxNews.com, Aljazeera.com

Google.com (perhaps not with external auth), Yahoo.com (incl email)

Wikipedia.org, Xkcd.com, www.acm.org (although not 'acm.org'), **not iee.org**

arin.net, ripe.net, apnic.net, lacnic.net, internet2.edu, aarnet.edu.au, nsf.gov,

isoc.org, csiro.au, es.net, ietf.org, nanog.org,

apple.com, microsoft.com, cleveland.com

applebees.com (but not locator map)

Juniper.com (with search), Cisco.com (no search), Brocade.com (no search),

www.a10networks.com, paloaltonetworks.com, jool.mx

(* includes streaming)



Poking around a NAT64 network

- - ping6 64:ff9b::128.171.6.64 (unices, mac os x)
 - ping 64:ff9b::128.171.6.64 (ms windows)
- You can actually exploit this, treating “64:ff9b::128.171.6.64” as an ipv6 address, as in:
 - [http://\[64:ff9b::128.171.6.64\]](http://[64:ff9b::128.171.6.64]) (linked on v6o)
 - Which uses the ipv4 address for v4o.tut.uhnet.net
- As long as you can reach a NAT64, and know the prefix you can use this
 - There are several public NAT64 trials



IPv4-As-A-Service: Alternatives

- To provide IPv4 connectivity to clients, you need to package IPv4 as a service
- There are various schemes, currently, including:
 - DNS64/NAT64
 - DS-Lite
 - MAP-T or MAP-E
 - IPv4 Residual Deployment (4rd)
 - etc



IPv4-As-A-Service: DS-Lite

- Provides access to v4 content over a native IPv6 network
- Provider handles tunneling over v6 from CPE to core
- V4 address is fake/NATed at core
- V6 is native
- Requires some end-host or CPE functionality



IPv4-As-A-Service

Stateful NAT64 + DNS64 + 464XLAT

Provides access to v4 content over a native IPv6 network

DNS64 causes v4 traffic to travel with mapped v6 addresses

464XLAT causes v4 traffic to get mapped locally

NAT64 handles translation of mapped v6 to v4 and NATs to v4 Internet

Does not require specific capabilities or config at end-user host

(Only DHCPv6 and/or RA-DNSS and/or static config)



Why we choose DNS64/NAT64

- It's the most implemented, mature, relevant scheme
- It does not require you to change end-user devices
- It uses regular, commodity hardware
- There are implementations of each piece available from multiple sources
- It collects globally routable IPv4 addresses in one place (or more for redundancy/load-sharing).
- Record of success (T-Mobile, others)
- Although the NAT64 holds state, it doesn't affect network topology
- To exclude it, you stop driving traffic to it.



NAT64 Is Not An In-Line Thing

- There's a tendency for documentation and tutorial content for NAT64 to depict it as a box through which all traffic flows.
- BUT the IPv4AAS construct is really the way to picture it -- it's a service that your network routes to, like it routes to anything else, and you drive traffic to it with DNS64



DNS64 (RFC6147)

- Comes as a feature in BIND, as well as other software
- Assumes the existence of a compatible IPv6<>IPv4 translator
- When it gets a request to resolve a name that has no AAAA=IPv6 response
 - It responds by synthesizing an AAAA response, encapsulating the A record response's IPv4 address into an IPv6 prefix, per RFC6052
 - Synthesis MUST be reversible
- MAY do reverse (PTR) answers for its own synthesized addresses
- Theoretically “breaks” DNSSEC, even though it doesn't “lie”
 - The number of end-host resolvers that ask for validation is very small
 - DNS64 clients are not currently designed to understand synthesis nor act on it
 - If they were, it's reversible, so the validation could be performed



DNS64 in practice

- Pretty simple to set up, if you've any BIND experience

```
options {
    directory "/var/cache/bind";
    dnssec-validation auto;
    auth-nxdomain no;      # conform to RFC1035
    listen-on-v6 { any; };
    allow-query { uhnet; };
    recursion yes;
    dns64 64:ff9b::/96 {
        clients {any;};
        mapped {!10/8; !172.16/12; any;};
    };
};
```



BIND9 DNS64 config

```
dns64 64:ff9b::/96 {  
    clients {any;};  
    mapped {!10/8; !172.16/12; any;};  
    #recursive-only yes;  
    #break-dnssec yes;  
  
};
```

- Clients - acl identifying clients to get DNS64 behavior from this server
- Mapped - acl specifying IPv4 range of A responses not to synthesize for
- Recursive-only - if this server has Authoritative zones, don't synthesize for those
- Break-dnssec - if a client requests DNS validation, ignore it and send responses regardless



DNS notes in general (BIND9 perspective)

- DNS64 can live on your main recursive server, or on a caching/forwarding server near your IPv6 only network
 - You can specify which clients get DNS64, others just get DNS
- DNS64 can serve from both IPv6 and IPv4 addresses (More about that later)
- You can put a DNS server on an IPv6 only network, but if it has no IPv4 connectivity, it needs to forward to a v4-connected server, because many domains don't have IPv6-connected name servers.

```
forwarders {  
    2001:0DB8::5353;  
    2001:0DB8::5:353;  
};  
forward only;
```



DNS64 And A Records

- Resolver behavior of happiness-oriented-browsers, legacy-yet-popular (Win7) OSes, etc may still choose A records if presented with both.
- DNS64 will still answer queries for A -- it only offers synthesized AAAA when it's asked for AAAA
- If the goal is to promote IPv6 use, and preserve IPv4-translation-address space, then it may be desirable to deny the existence of the A record.
- A record suppression has come up in previous test network scenarios, but usually to address user experience issues with Happy Eyeballs, etc, not to maximize IPv4 address use



NAT64: NAT That Shrinks, Rather Than NAT That Grows

- Better than NAT44 for various reasons:
 - Begins as an enhancement to a v6 network
 - The more IPv6 grows, the less NAT you have to do.
 - Promoting Native IPv6 reduces use of NAT v4 address/port space
- Better than dual-stack because the more you shift to IPv6 only networking, the less you spend precious IPv4 addresses on building an IPv4 network
- As IPv4 addresses become more expensive and more scarce, providing native IPv4 addresses on per-interface assignments becomes unthinkable.



NAT64

- Accepts packets destined for translation-prefixed destinations, translates them to IPv4
- Accepts response packets from IPv4 destinations, sends them to initiator, with translation-prefix source addresses



Stateless NAT64 (SIIT)

- Stateless: 1:1 address mapping, appropriate for making your IPv4 servers look like they're doing IPv6
 - Probably most often superseded by simply turning on IPv6 on servers.
 - May become a glue-technology for long-life IoT, like power, HVAC, telecom management



Stateful NAT64

- Stateful: N:1 NAT providing IPv4 As A Service to an IPv6-only network
- Uses address mappings in the form of:
 - IPv4: 198.51.100.18
 - Synthesized IPv6: 64:ff9b::198.51.100.18
 - a.k.a. 64:ff9b::c633:6412
- (God Bless whomever came up with the append-dotted-decimal notation)



Purpose-built Routers vs. Linux Routers

- We have 2 active NAT64 running at U. Hawaii
 - Cisco ASR1001, running IOS-XE
 - Version 15.3(2)S2, RELEASE SOFTWARE (fc1) - IOS XE Version: 03.09.02.S
 - Documentation isn't really plentiful, as far as I can see.
 - *Probably* more scalable than a Linux router, just a guess
 - Ubuntu 16.04 with Jool NAT64
 - Jool 3.5.4 (<http://jool.mx>)
 - Exhaustively documented, more knobs, easier to monitor and tweak



IOS-XE

```
interface GigabitEthernet0/0/1
  description nat64 v4 side
  ip address 203.0.113.190 255.255.255.252
  ip ospf authentication-key 7 283947219834821734
  ipv6 nd ra suppress all
  nat64 enable
!
interface GigabitEthernet0/0/2
  description nat64 v6 side
  no ip address
  ipv6 address 2001:0DB8:4101:FF::1/64
  ipv6 enable
  nat64 enable
  ipv6 ospf 1 area 0
```



IOS-XE

```
ipv6 router ospf 1
  redistribute static << NAT64 inserts routes as statics
```

```
ipv6 access-list nat64-acl
  permit ipv6 2001:0DB8::/32 any
```

```
nat64 logging translation flow-export v9 udp destination 198.51.100.105 10001
nat64 v4 pool pool1 203.0.113.240 203.0.113.255
nat64 v6v4 list nat64-acl pool pool1 overload
```

^^ logging occurs as flow export - nfdump reads it like:

```
1969-12-31 14:00:00.000 0.000 ICMP 2607:f278:0:2::14.0 -> :::0.0 ..... 0 0 0 1
```



IOS-XE

```
NAT64#sho ipv6 route stat
IPv6 Routing Table - default - 683 entries
Codes: C - Connected, L - Local, S - Static, U - Per-user Static route
       B - BGP, R - RIP, H - NHRP, I1 - ISIS L1
       I2 - ISIS L2, IA - ISIS interarea, IS - ISIS summary, D - EIGRP
       EX - EIGRP external, ND - ND Default, NDp - ND Prefix, DCE - Destination
       NDr - Redirect, O - OSPF Intra, OI - OSPF Inter, OE1 - OSPF ext 1
       OE2 - OSPF ext 2, ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
       ls - LISP site, ld - LISP dyn-EID, a - Application
S   64:FF9B::/96 [1/0]                << I did not type this in as a static route
    via ::100.0.0.2, NVI0              (or even as a NAT64 prefix, apparently it's default/implicit)
S   2001:0DB8:F0:0:1::/96 [1/0]       << nor did I enter this static (but it I did enter nat64 prefix)
    via ::100.0.0.1, NVI0
```



IOS-XE

```
NAT64#sho nat64 statistics global
NAT64 Statistics
```

```
Total active translations: 22 (0 static, 22 dynamic; 22 extended)
Sessions found: 117305872
Sessions created: 594195
Expired translations: 594291
Global Stats:
  Packets translated (IPv4 -> IPv6)
    Stateless: 0
    Stateful: 81459865
    MAP-T: 0
  Packets translated (IPv6 -> IPv4)
    Stateless: 0
    Stateful: 36440237
    MAP-T: 0
```



Stateful NAT64 on Linux: Jool 3.54

- <http://jool.mx>
- Pronounced like English “jewel”, it is a Mayan word, not a Spanish word
- I have set up Jool in Ubuntu 16 & Raspian 8, to good effect
- It is the only Linux based SIIT/NAT64 solution I have tried
 - It is currently actively developed, with attention on functionality and performance, and seems to be best-of-breed



NAT64 on Linux: Jool 3.54

- One needs to prep Ubuntu like a router/translator
- You can do one-armed NAT64 pretty easily
- Following assumes separate v6 interface and v4 interface



Sysctl.conf

```
(...)  
net.ipv4.conf.all.forwarding=1  
net.ipv6.conf.all.forwarding=1  
  
# limit ephemeral ranges to add more ports to translation range  
net.ipv4.ip_local_port_range = 1025 1125
```



interfaces

```
auto ens19
iface ens19 inet6 static
  address 2001:0db8:F101:6001::64
  netmask 64
  accept_ra 1
  autoconf 0
  privext 0
```

```
auto ens18
iface ens18 inet static
  address 203.0.213.146
  netmask 255.255.255.240
  gateway 203.0.213.81
```

```
auto ens18:147
iface ens18:147 inet static
  address 203.0.213.147
  netmask 255.255.255.240
```

```
auto ens18:148
iface ens18:148 inet static
  address 203.0.213.148
  netmask 255.255.255.240
```

```
# (...)
```



Skip the broadcast

DON'T configure the IP broadcast addr as a pool address.

Jool will happily use the segment broadcast as a source address, which works, unfortunately, but as a destination address, the router gets all particular about delivering to broadcast



rc.local

```
#This is required once you enable ipv6 forwarding
# it probably belongs in /etc/network/interfaces
/bin/ip -6 route add default via 2001:0DB8:F101:6001::1
# per http://jool.mx/en/run-nat64.html
# and http://jool.mx/en/offloads.html
/sbin/ethtool --offload ens18 gro off
/sbin/ethtool --offload ens18 lro off
/sbin/ethtool --offload ens18 gso off
/sbin/ethtool --offload ens18 tso off
/sbin/ethtool --offload ens18 tx off
/sbin/ethtool --offload ens18 rx off
#( and again for ens19 )
/sbin/modprobe jool pool6=2001:0db8:f0:0:2::/96
/usr/local/bin/jool --pool4 --add 203.0.113.147 1126-65535
/usr/local/bin/jool --pool4 --add 203.0.113.148 1126-65535
/usr/local/bin/jool --pool4 --add 203.0.113.149 1126-65535
/usr/local/bin/jool --pool4 --add 203.0.113.150 1126-65535
/usr/local/bin/jool --pool4 --add 203.0.113.151 1126-65535
/usr/local/bin/jool --pool4 --add 203.0.113.152 1126-65535
/usr/local/bin/jool --pool4 --add 203.0.113.153 1126-65535
(...)
```



Userspace jool command to monitor

```
/usr/local/bin/jool --pool4 --display
```

```
0      TCP      203.0.213.147  1126-65535
0      TCP      203.0.213.148  1126-65535
0      TCP      203.0.213.149  1126-65535
(...)
0      UDP      203.0.213.147  1126-65535
0      UDP      203.0.213.148  1126-65535
0      UDP      203.0.213.149  1126-65535
(...)
0      ICMP     203.0.213.147  1126-65535
0      ICMP     203.0.213.148  1126-65535
0      ICMP     203.0.213.149  1126-65535
(...)
(Fetched 36 samples.)
```



NAT64 is a service on the network

- I export the routes into our OSPF and then *any IPv6-connected host* can use it
 - Pinging an IPv4 address by hand-mapping it into a NAT64 prefix:

```
[root@robespierre ipv6]# ping6 -c 5 64:ff9b::203.0.213.116
PING 64:ff9b::203.0.213.116(64:ff9b::cb00:d574) 56 data bytes
64 bytes from 64:ff9b::cb00:d574: icmp_seq=1 ttl=56 time=1.64 ms
64 bytes from 64:ff9b::cb00:d574: icmp_seq=2 ttl=56 time=1.44 ms
64 bytes from 64:ff9b::cb00:d574: icmp_seq=3 ttl=56 time=1.46 ms
64 bytes from 64:ff9b::cb00:d574: icmp_seq=4 ttl=56 time=1.20 ms
64 bytes from 64:ff9b::cb00:d574: icmp_seq=5 ttl=56 time=2.12 ms

--- 64:ff9b::203.0.213.116 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4008ms
rtt min/avg/max/mdev = 1.201/1.574/2.124/0.310 ms
```

- Of course, one might need to think about limiting access



Neither NAT64 Can Reach Its Own Translator

- From the command line on either the Cisco ASR 1001 OR on the Ubuntu/Jool NAT64
 - Pinging 64:ff9b::203.0.213.13 results in no responses.



Abracadabra

- With a NAT64 active and routing for the translation prefix set up
- On a v6 only host, and you need to ssh to a v4-only host

```
whinery@bitwclat:~$ ssh 64:ff9b::203.0.213.64
whinery@64:ff9b::203.0.213.64's password:
Welcome to Ubuntu 16.04.2 LTS (GNU/Linux 4.4.0-87-generic x86_64)
*** System restart required ***
Last login: Tue Oct 10 14:55:35 2017 from 203.0.213.99
whinery@v4-only:~$
```



Scenario 2: IPv6-only with DNS64/NAT64

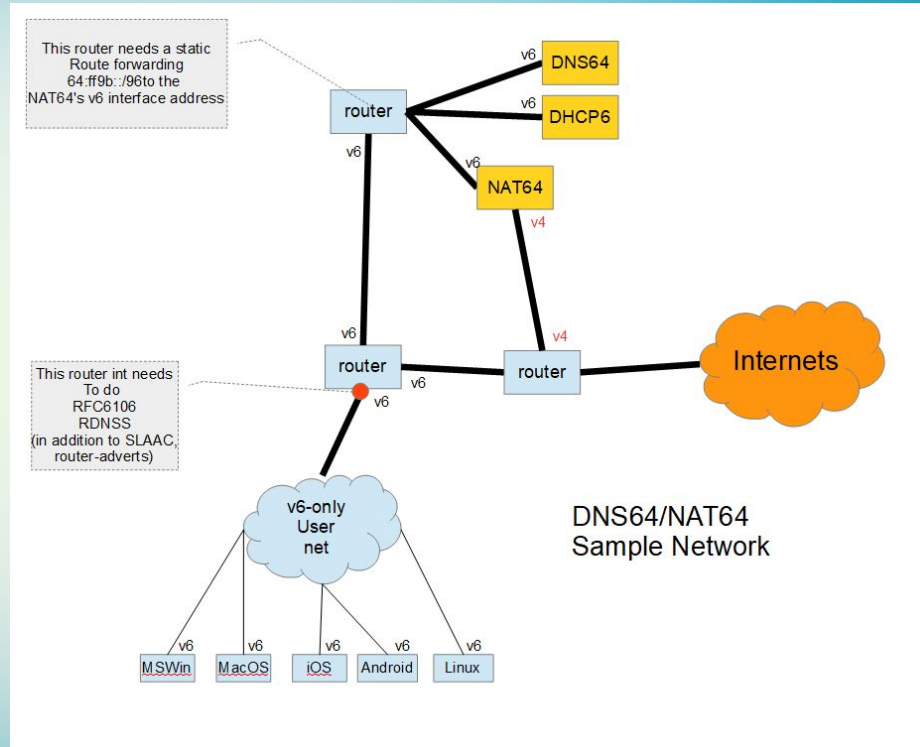
Taking v6-only from scenario 1

Adding NAT64 translator

Turning on DNS64 on the DNS server

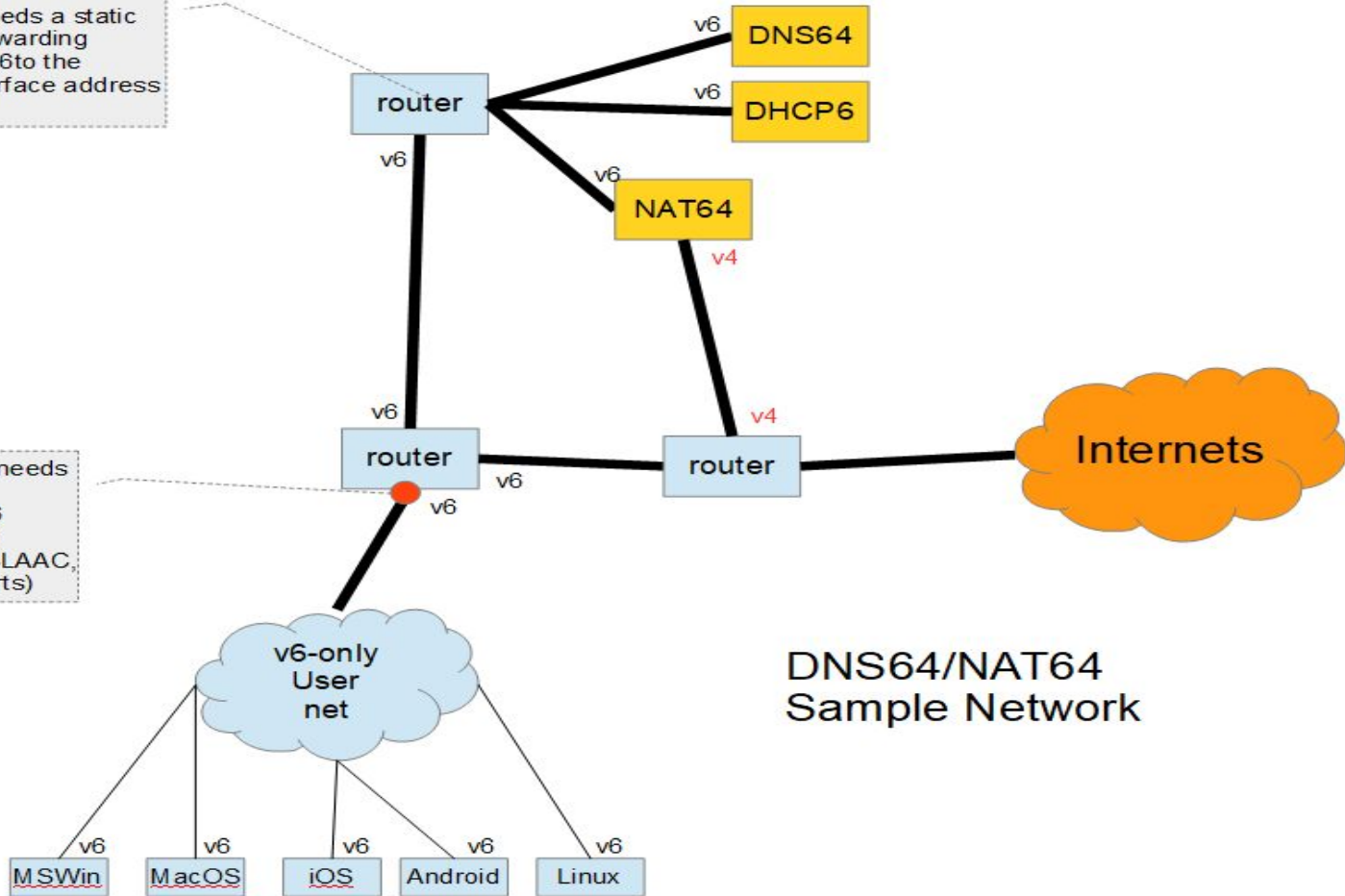
Confirming a static route to NAT64 for the translation prefix

This network will be much more versatile than experiment 1, but eventually you will find something that does not work.



This router needs a static Route forwarding 64:ff9b::/96 to the NAT64's v6 interface address

This router int needs To do RFC6106 RDNSS (in addition to SLAAC, router-adverts)

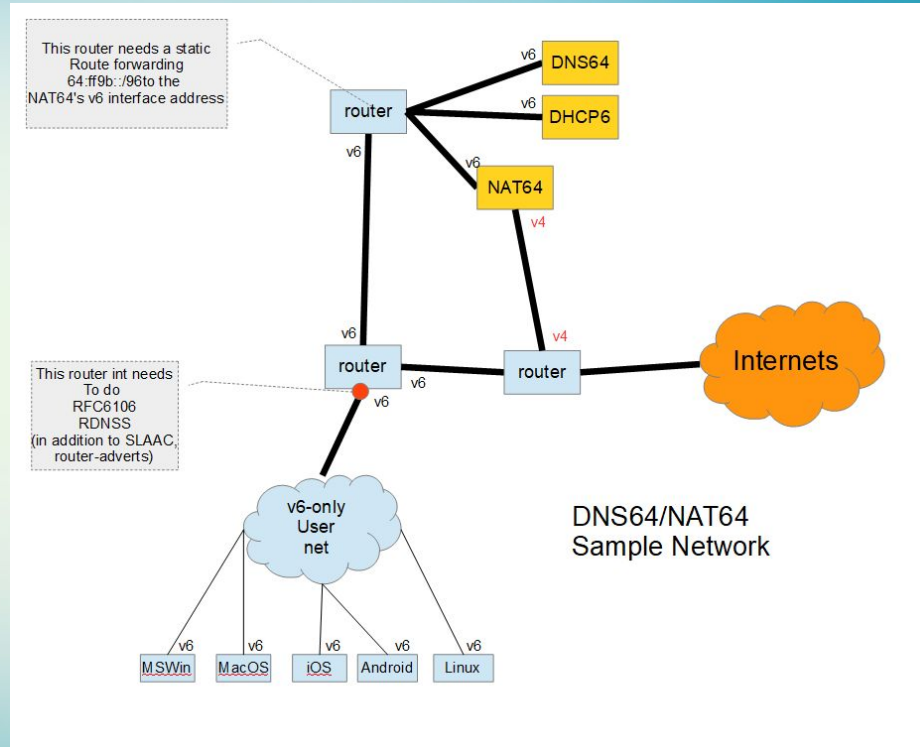


DNS64/NAT64
Sample Network

Scenario 2: IPv6-only with DNS64/NAT64

- Load web page:
 - <http://v4o.tut.uhnet.net>
- If the left-hand picture fails to load, then IPv4 literals are not working.

*Several modern oses should be able to reach v4 literals because they have built-in IPv4 literal-catchers (aka CLAT, or 464XLAT)



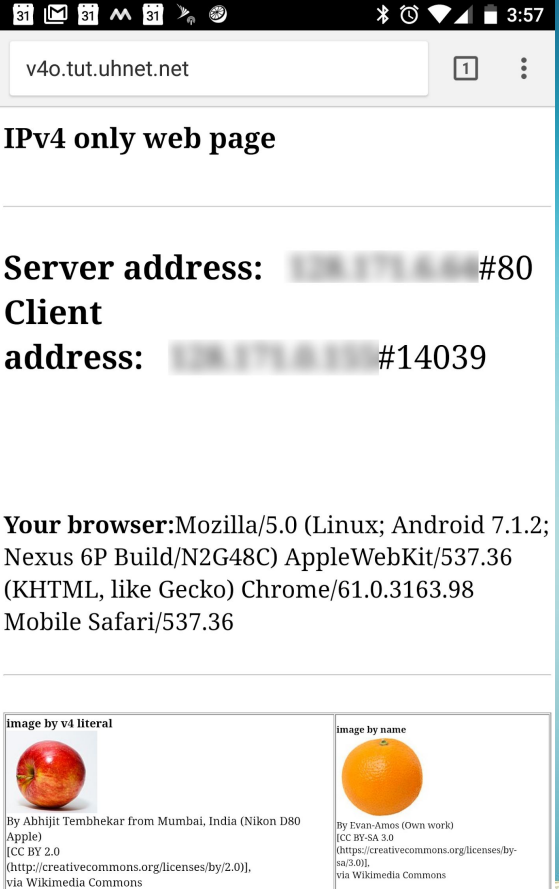
Scenario 2: IPv6-only with DNS64/NAT64

- Note that there are still no IPv4 addresses at the clients
- If you encounter failures, they are most likely caused by “IPv4 Literals”, which are numeric IPv4 addresses embedded in HTML, or software code. Since numeric addresses do not require an address lookup, they’re not re-synthesized by DNS64.



Devices That Get The Apple

- Why?
- Because they have built-in 464XLAT CLAT.
- When such a device joins a v4-only network, it queries the AAAA record for “v4only.arpa”
- Since it knows the expected response, and can deduce the prefix from the AAAA response, it knows about DNS64 and NAT64, including the synth/translation prefix



The screenshot shows a mobile browser interface. At the top, the address bar contains 'v4o.tut.uhnet.net'. Below the address bar, the page title is 'IPv4 only web page'. The main content area displays the following information:

- Server address:** [redacted] #80
- Client address:** [redacted] #14039
- Your browser:** Mozilla/5.0 (Linux; Android 7.1.2; Nexus 6P Build/N2G48C) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/61.0.3163.98 Mobile Safari/537.36

At the bottom of the page, there are two image thumbnails:

- image by v4 literal:** An image of a red apple. Below it, the text reads: 'By Abhijit Tembhekar from Mumbai, India (Nikon D80 Apple) [CC BY 2.0] (<http://creativecommons.org/licenses/by/2.0>), via Wikimedia Commons'.
- image by name:** An image of an orange. Below it, the text reads: 'By Evan-Amos (Own work) [CC BY-SA 3.0] (<https://creativecommons.org/licenses/by-sa/3.0/>), via Wikimedia Commons'.



464XLAT (RFC6877)

- 464XLAT is a superset of NAT64, and it can be a transition strategy in itself.
- Uses a simple form of SIIT NAT46 to clean up residual IPv4 requests from software, such as hard-coded IPv4 literals, and send them to NAT64
 - Employs 2 pieces:
 - CLAT (at the client)
 - PLAT (at the provider, usually will be the NAT64)
- Takes stray IPv4 packets, translates them, sends them to NAT64
- Has 3 styles:
 - Bump-in-the-wire
 - Meaning there's a local device that does the duty
 - Bump-in-the-host (bump-in-the-stack)
 - Meaning that there's a feature in the end host that translates stray packets before they hit the wire
 - Bump-in-the-browser



People don't say "464CLAT"

- 464XLAT is a superset of NAT64
- It involves a CLAT (NAT46 at the client)
- And a PLAT (NAT64 at the "provider")
- But since we just got done talking about NAT64 and we're moving to 464XLAT talk, I hereby coin "464CLAT" use it in good health, no need to thank me...



Experiment 3: Adding 464CLAT

- Using the `jool_siit` module, we put a NAT46 on the IPv6-only wire
- The NAT46 offers RFC1918 addresses in DHCPv4 to the hosts on the wire, and poses as the local v4 gateway
- Clients send packets to the specified CLAT, which takes the packets and translates them to IPv6 to sent to PLAT (NAT64)



464XLAT (AKA 464CLAT) with Jool_SIIT

/etc/sysctl.conf

```
#(...) append the following to the end of existing /etc/sysctl.conf
# per http://jool.mx/en/run-nat64.html
net.ipv4.conf.all.forwarding=1
net.ipv6.conf.all.forwarding=1
```

/etc/rc.local

```
#(...)
ip route add default via 2607:f278:4101:6002::1
ethtool --offload ens18 gro off
ethtool --offload ens18 lro off

# pool6 is the dest of the NAT64 translation pool
modprobe jool_siit pool6=64:ff9b::/96
# EAMT shows the fake v4 sources, and maps to v6 sources to reach the NAT64 pool6 destinations.
jool_siit --eamt --add 172.31.240.0/20 2001:0db8:f0:0:3::/96

exit 0
```



464XLAT (AKA 464CLAT) with Jool_SIIT

/etc/dhcp/dhcpd.conf

```
# this file is for a DHCP4 running on a 464XLAT CLAT, which acts as a
# NAT64 to relay ipv4 packets to NAT64

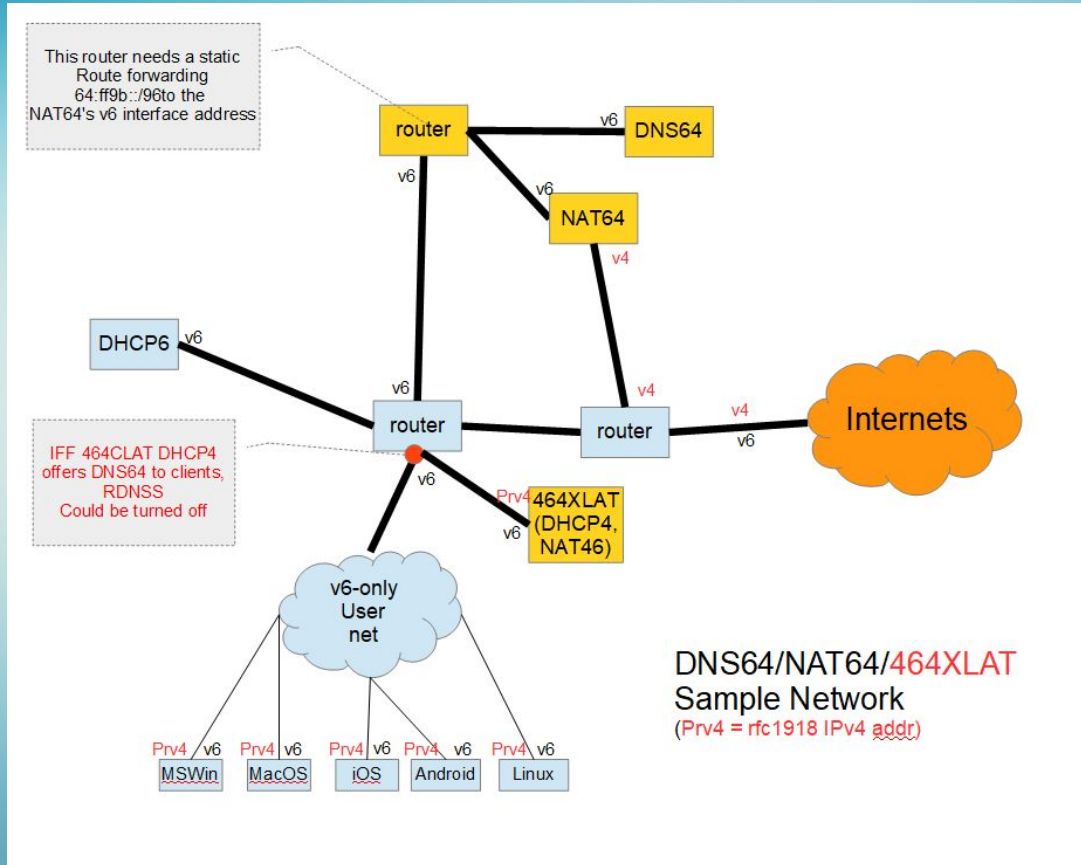
default-lease-time 600;
max-lease-time 7200;

# Use this to send dhcp log messages to a different log file (you also
# have to hack syslog.conf to complete the redirection).
log-facility local7;

subnet 172.31.240.0 netmask 255.255.240.0 {
    range 172.31.240.20 172.31.255.254;
    # NEVER ANY DNS server entries here.
    ##### option domain-name-servers ns1.internal.example.org;
    ##### option domain-name "internal.example.org";
    option subnet-mask 255.255.255.224;
    option routers 172.31.240.1;
    option broadcast-address 172.31.255.255;
    default-lease-time 600;
    option interface-mtu 1460;
    max-lease-time 7200;
}
```



Scenario 3: Adding 464CLAT



Implications Of 464CLAT

- IPv4-only devices work
- IPv4 literals and v4-preferring resolvers work
- While we have left IPv6 DNS adverts from the router active for experiment 3, it's no longer needed to support Android
 - If you provide an IPv4 address for the DNS64 on the 464CLAT DHCP, it works
 - If the DNS is not on the wire, you will use translation space on DNS queries
 - If running SLAAC is a problem for you, you can turn it off and let Android devices just use NATted v4.
 - Using DHCP6 in lieu of SLAAC is a very common



Implications Of 464CLAT

- Support for RDNSS on current routers is somewhat limited.
 - Cisco supports half of RFC6106 - RDNSS, but no DNSSEC
 - You need to have devices that run relevant IOS-XE
 - “Regular” IOS is left out
 - Juniper appears to have similar product line constraints
 - So the answer for many to have RDNSS is currently “buy a new router”
- So if you’re in the CLAT for stray-IPv4-literal business anyways, you may as well put a forwarding resolver to DNS64 on the CLAT



Stateful NAT64 Manufacturer Support

- Cisco In IOS-XE on certain router platforms
- Cisco ASA (incl DNS64)
- Juniper routers (MX,SRX,?)
- Microsoft Forefront Unified Access Gateway (incl DNS64)
- Palo Alto
- A10 Networks (incl DNS64)
- F5



But Wait, There's More

- ARIN constituents who intend to employ IPv6 transition technologies can (pretty easily) receive an IPv4 block, between /28 and /24 inclusive, to use for that purpose
- ARIN set aside an IPv4 /10 for this, described in the Number Resource Policy Manual (NRPM) coincidentally-numbered section 4.10
- <https://www.arin.net/policy/nrpm.html#four10>
- In NAT64, 256 addresses would support (on the order of) 16 million NAT mappings



Enhancing the Network with IPv6

- PvDs (Provisioning Domains)
 - More intelligence to the client node for:
 - Multi-homing
 - Service selection
 - Detecting Captive Portal and other things on the network.
- Segment routing
 - Embedding paths in the headers
 - Source chooses the path
 - Per application



Interesting Linux Routers

- Linux Routers:
 - VPP (https://wiki.fd.io/view/VPP/NAT#Stateful_NAT64)
 - Any Linux box with routing software



End



L3 Translation: MTU, Fragmentation, and PMTUD

IPv4 (typically) derives MSS from MTU - 40

IPv6 (typically) derives MSS from MTU - 60

When a v4-origin TCP segment needs to be translated to a v6-destined TCP segment, Somebody needs to provide 20 bytes worth of change.

V4 PMTUD works with “fragmentation needed” messages

V6 PMTUD works with “packet too big (hulk smash)” messages



L3 Translation: MTU, Fragmentation, and PMTUD

Turning on

```
Net.ipv4.tcp_mtu_probing on linux server works
```



Ephemeral source port ranges

When using Linux boxes as routers -

One needs to track possibility of port collisions

If you're going to do scale, you need free ports to represent flows

Stateful NAT64 uses 5-tuple (proto,src, srcprt,dst,dstprt) similar to NAT-PT

Stateless SIIT EAMT (for 464 CLAT) does not use ports, only direct mappings, so SIIT box port range simply needs to stay clear of client ranges



APNIC NAT64 Checker

<https://blog.apnic.net/2017/08/23/introducing-nat64-checker/>



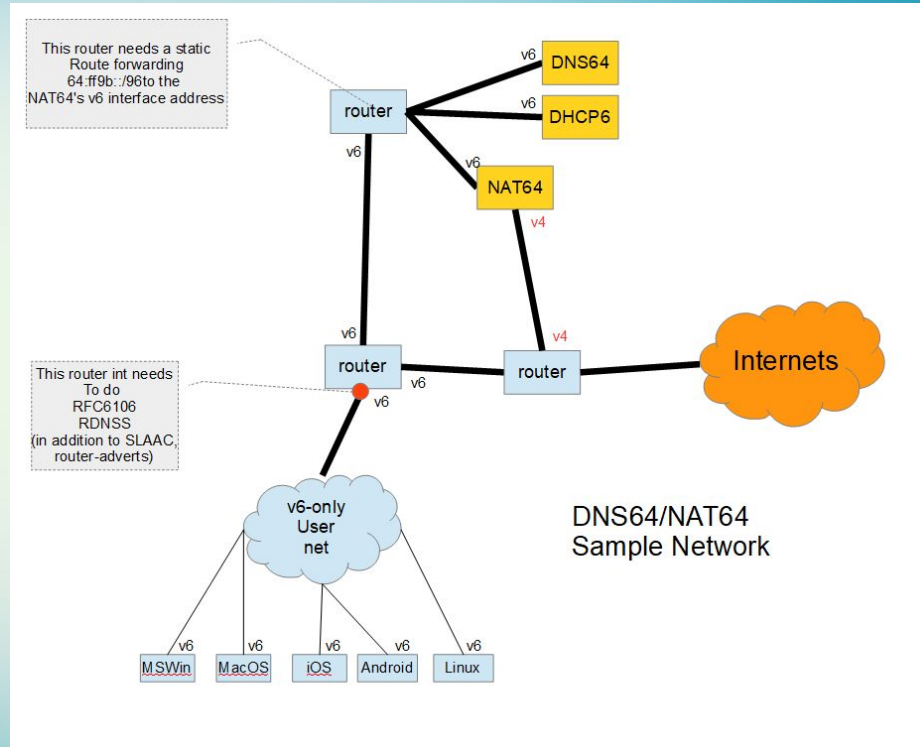
Take A Few Minutes And Explore

- 1) Get connected and try
 - a) <http://v6o.tut.uhnet.net> (which should work)
 - b) <http://v4o.tut.uhnet.net> (which should *not* work)
 - c) ping6 www.google.com (MS-Windows: “ping www.google.com”)
 - d) If no luck, check IPv6 address, DNS server assignment, etc.
- 2) What color is your IPv6?
 - a) Find out here: <https://www.vyncke.org/myipcolor.php> (linked on v6o)
- 3) Streaming video the IPv6 way:
 - a) <http://demo.6cn.solutions/users/play.php?id=4&cdn=> (linked on v6o)



Experiment 2: IPv6-only with DNS64/NAT64

- Try (mac/linux/android)
 - “ping6 v4o.tut.uhnet.net”
- Try (windows)
 - “ping v4o.tut.uhnet.net”
- Try (mac/linux/android)
 - “ping v4o.tut.uhnet.net”
- Try (windows)
 - “ping -4 v4o.tut.uhnet.net”



Experiment 3: Adding 464CLAT

- You should un-join/re-join the SSID, so that your client does DHCP4 now
- You may be hard-pressed to find things that don't work with this network
 - To the extent that I wonder whether it isn't at least equivalent, in terms of percent functionality with NAT44 CGN

