



NATNATNAT & some NS-Set Crumbs:

Measuring DNS Resolvability with NAT64 & Minimal NS Sets

Tobias Fiebig, Zilan Cheikho, Taha Albakou & Anja Feldmann

Max-Planck Institut für Informatik



- Measurements on IPv6 resolvability and IPv6/Fragmentation
- Paper: 'How I learned to stop worrying and love IPv6': Measuring the Internet's Readiness for DNS over IPv6. IMC '25.
- Presentation @IETF123 DNSOP; Work on RFC3901bis
- Presentation @IETF122 IEPG
 - Jen: *"But what about NAT64?!"*





- Measurements on IPv6 resolvability and IPv6/Fragmentation
- Paper: ‘How I learned to stop worrying and love IPv6’: Measuring the Internet’s Readiness for DNS over IPv6. IMC ’25.
- Presentation @IETF123 DNSOP; Work on RFC3901bis
- Presentation @IETF122 IEPG
 - Jen: *“But what about NAT64?!”*

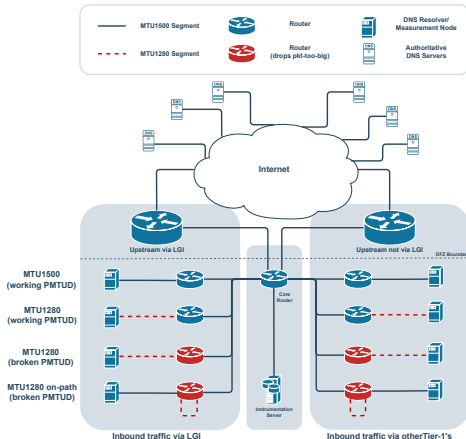


Recap: Measuring DNS Resolution



- Resolve the Google CrUX Top 10M IPv4 only/IPv6 only/dual-stack
- Use a 'minimally covering NS set', i.e., hit each NS Set once (1x DNSSEC, 1x without DNSSEC)
- Implement various MTU scenarios:
 - MTU1500
 - MTU1280 on-link (with working PMTUD)
 - MTU1280 on-link (with broken PMTUD)
 - MTU1280 on-path (with broken PMTUD)

! All data is available on
<https://dns-mtu.measurement.network/> (past 7 days)
and <https://data.measurement.network/>



NAT64 Setup



- Added during IETF122;
 - Add 4x Measurement host
 - Add 4x NAT64 host (OpenBSD)
 - Measure
 - Get annoyed emails for sending too many TCP RST
 - Thanks & apologies to Sven van Dyck and Geert Verheyen from *dns_belgium*
 - Why? Out of state table? OpenBSD just slow?
- Hand out thesis and find out which NAT64 solution we should use



NAT64 Setup



- Added during IETF122;
 - Add 4x Measurement host
 - Add 4x NAT64 host (OpenBSD)
 - Measure
 - Get annoyed emails for sending too many TCP RST
 - Thanks & apologies to Sven van Dyck and Geert Verheyen from *dns_belgium*
 - Why? Out of state table? OpenBSD just slow?
- Hand out thesis and find out which NAT64 solution we should use



NAT64 Setup



- Added during IETF122;
 - Add 4x Measurement host
 - Add 4x NAT64 host (OpenBSD)
 - Measure
 - Get annoyed emails for sending too many TCP RST
 - Thanks & apologies to Sven van Dyck and Geert Verheyen from *dns_belgium*
 - Why? Out of state table? OpenBSD just slow?
- Hand out thesis and find out which NAT64 solution we should use



NAT64 Setup



- Added during IETF122;
 - Add 4x Measurement host
 - Add 4x NAT64 host (OpenBSD)
 - Measure
 - Get annoyed emails for sending too many TCP RST
 - Thanks & apologies to Sven van Dyck and Geert Verheyen from *dns_belgium*
 - Why? Out of state table? OpenBSD just slow?
- Hand out thesis and find out which NAT64 solution we should use



NAT64 Setup



- Added during IETF122;
 - Add 4x Measurement host
 - Add 4x NAT64 host (OpenBSD)
 - Measure
 - Get annoyed emails for sending too many TCP RST
 - Thanks & apologies to Sven van Dyck and Geert Verheyen from *dnsbelgium*
 - Why? Out of state table? OpenBSD just slow?
- Hand out thesis and find out which NAT64 solution we should use



NAT64 Setup



- Added during IETF122;
 - Add 4x Measurement host
 - Add 4x NAT64 host (OpenBSD)
 - Measure
 - Get annoyed emails for sending too many TCP RST
 - Thanks & apologies to Sven van Dyck and Geert Verheyen from *dns_belgium*
 - Why? Out of state table? OpenBSD just slow?
- Hand out thesis and find out which NAT64 solution we should use



NAT64 Setup



- Added during IETF122;
 - Add 4x Measurement host
 - Add 4x NAT64 host (OpenBSD)
 - Measure
 - Get annoyed emails for sending too many TCP RST
 - Thanks & apologies to Sven van Dyck and Geert Verheyen from *dns_belgium*
 - Why? Out of state table? OpenBSD just slow?
- Hand out thesis and find out which NAT64 solution we should use



Picking the Right NAT64 Solution

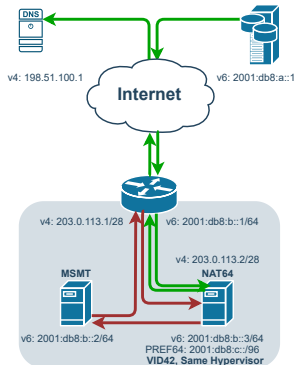


- Thesis by Zilan Cheikho (HTW Saar / Max-Planck-Institut für Informatik)
- Benchmark Tayga, Jool, and OpenBSD
- Evaluate DNS traffic `TIMEOUTs` @MTU1500, small/big packet bandwidth, with and without offloading
- Base setup similar to 'production' measurements: Minimally covering NS-Set, ZDNS, Unbound



Too many TCP RST

- NAT64 works from remote
- Not from the msmt. host
 - ICMPv4 → works
 - UDP → Port Unreachable
 - TCP → Immediate RST
- Happens for Jool, Tayga, and OpenBSD alike
- TCP RST sent by NAT64 host
- ?!



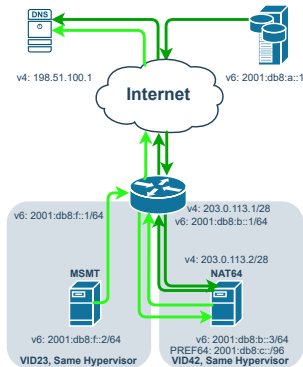
- Downloaded from <https://www.cambridge.org/core>. University of Cambridge, on 02 Jun 2020 at 10:00:00, subject to the Cambridge Core terms of use, available at <https://www.cambridge.org/core/terms>. <https://doi.org/10.1017/S0022216X20000539>



Too many TCP RST



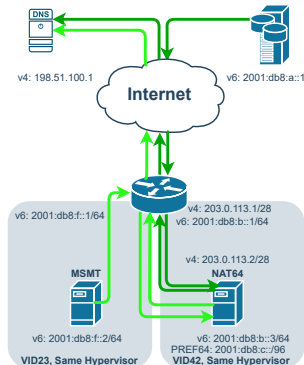
- NAT64 works from remote
- Not from the msmt. host
 - ICMPv4 → works
 - UDP → Port Unreachable
 - TCP → Immediate RST
- Happens for Jool, Tayga, and OpenBSD alike
- TCP RST sent by NAT64 host
- Works without shared VLAN



Too many TCP RST



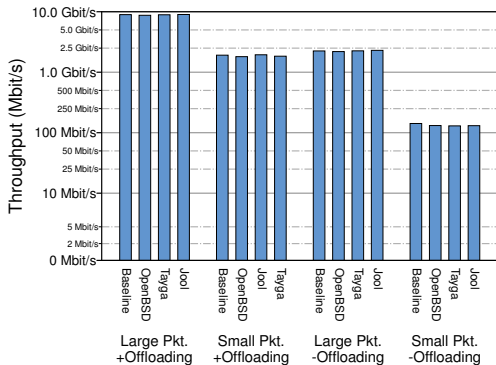
- Bug in Linux/KVM/Virtio offloading?
- Bug in HW NIC offloading?
No offloading → no help. o.O
- PCAPs available!



NAT64 Measurement Results



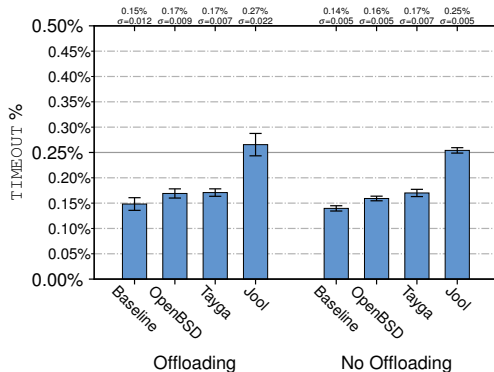
- iperf3 over 5 minutes on a 10gbit path
- Throughput is better with offloading for all three, close to native
- Apart from that, no major differences; OpenBSD and Tayga a tad behind Jool



NAT64 Measurement Results



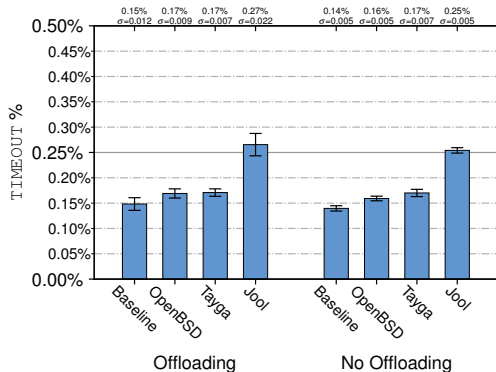
- Throughput is better without offloading for all three, close to native
- Tayga & OpenBSD are close to the baseline for DNS, Jool is a bit behind
- No offloading reduces noise
- Using OpenBSD seems fine



NAT64 Measurement Results



- Throughput is better without offloading for all three, close to native
- Tayga & OpenBSD are close to the baseline for DNS, Jool is a bit behind
- No offloading reduces noise
- Using OpenBSD seems fine



Minimally Covering NS Sets



- Idea from Petr Špaček at IETF121: Hit each NS Set only once
 - Reduces load
 - Should still hit all oddities
- Problems:
 - Some really dominant NS Sets (GoDaddy, Cloudflare, domain parking)
 - Operators with many similar NS Sets (Cloudflare, Amazon)
- **Question:** Is this really representative?
- Project with Taha Albakou



Methodology



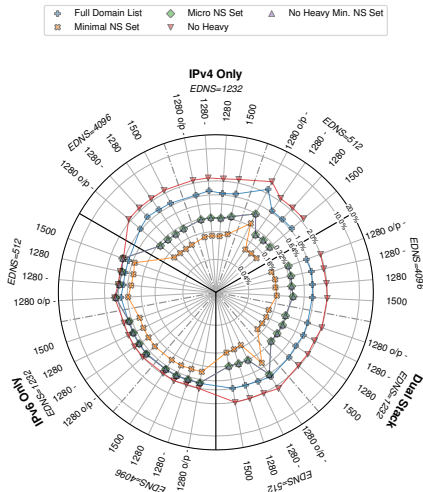
- Measure the Google CrUX Top 10M without minimally covering NS Sets
- Scale dataset based on what the minimally covering NS Set would have looked like
- Re-Scale from minimally covering NS Set weighted based on the number of domains
- Slices:
 - Minimally Covering NS Set (one zone per NS Set)
 - Micro Covering NS Set (aggregate heavy hitter NS Sets, CF/Amazon etc.)
 - No Heavy Hitters (Remove Top NS Sets)
 - Minimally Covering NS Set over No Heavy Hitters set



Results: Without DNSSEC



- Relatively, *overestimates* IPv6 TIMEOUTs and *underestimates* IPv4 TIMEOUTs
- Removing top-NS Sets makes everything worse (top NS Sets introduce positive bias)
- IPv6 looks, overall, better
- Dual-Stack seems to be mostly influenced by IPv4



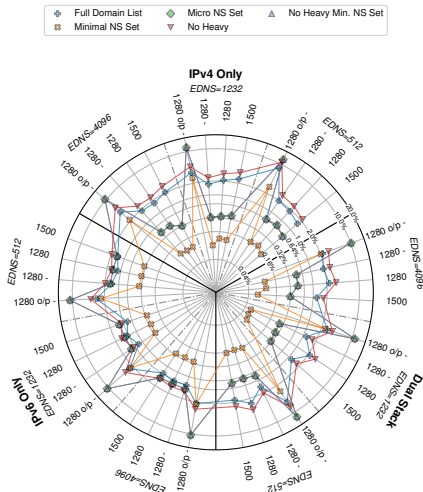
- 



Results: With DNSSEC



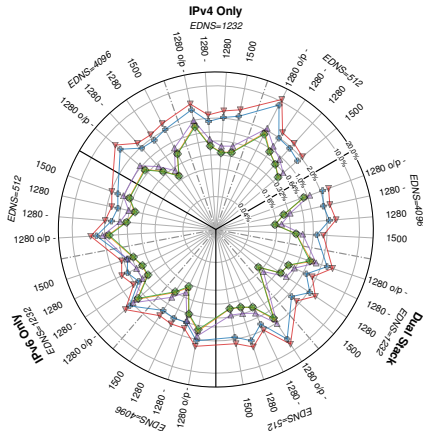
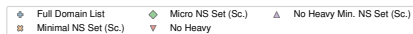
- High TIMEOUTs for MTU1280 on-path with broken PMTUD regardless of IPv4/IPv6/Dual-Stack (over-amplified in minimal NS-Set)
- IPv6 looks worse in minimal NS Sets than over the full set
- Again, the overall pattern remains similar



Results: With DNSSEC (Scaled)



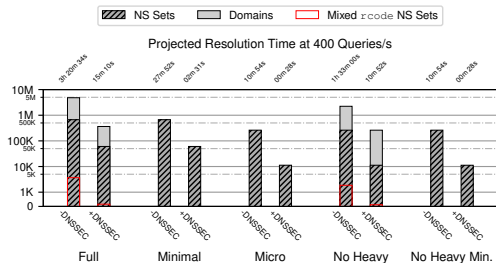
- Overall less difference between scaled/full sets than without DNSSEC
- IPv6 again closer to scaled values
- Under-Estimation of TIMEOUTs for 1280MTU on-link in minimal sets



Minimal NS Sets: Bottom Line



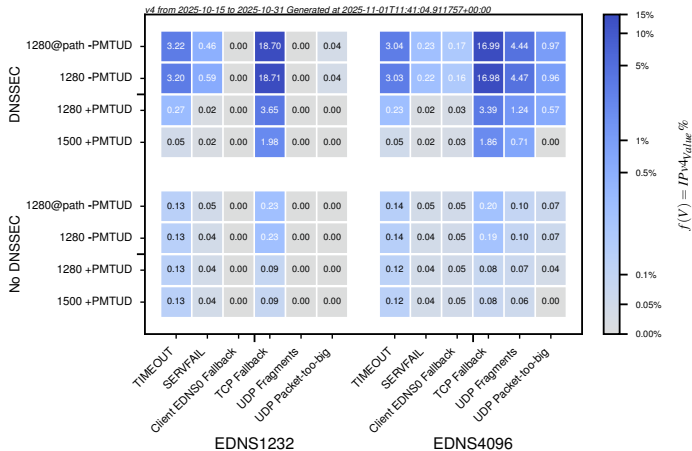
- Minimally covering NS Sets / Micro covering NS Sets save a lot of time when measuring
- A fraction of NS Sets (around 0.1%) has 'mixed' rcodes
- Scaling minimal NS Sets gets closer to the absolute values to expect
- Relative* values are mostly fine (IPv6 has it a bit harder, though)



NAT64 (with a Minimally Covering NS Set... Set)



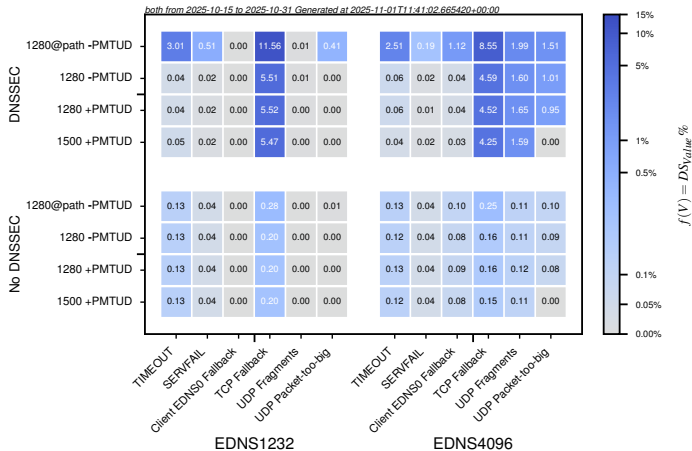
- Timeout baseline for 'No DNSSEC' (likely from forced DualStack)
- Less TIMEOUTs w. DNSSEC @MTU1500
- Broken PMTUD is TCP Fallback heavy
- Difference between an on-path MTU break and an on-link break is limited



Dual-Stack



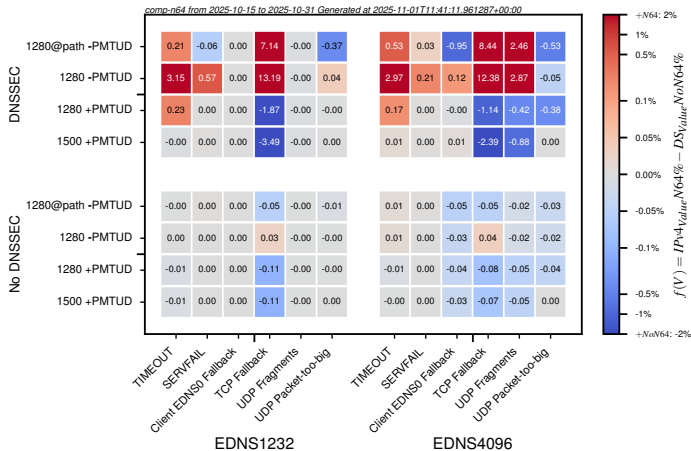
- DS also suffers for on-path breaks without PMTUD
- Similar TIMEOUT baseline without DNSSEC
- Overall consistent with prior findings for v4/v6



NAT64 vs. Dual-Stack



- Comparable @MTU1500, and generally without DNSSEC
- OK-ish similar for on-path MTU break without PMTUD & on-link with PMTUD
- Worse for on-link break without PMTUD (ca. similar to on-path in class)



Conclusion



- Offloading... a story... full of misunderstandings...
- Minimally covering NS Sets work (but better scale them back up if you are looking for absolute values)
- Better skip on NAT64 for DNS and try to go native

Data: <https://data.measurement.network>

(Ask me for NFS access if you want to copy everything)

