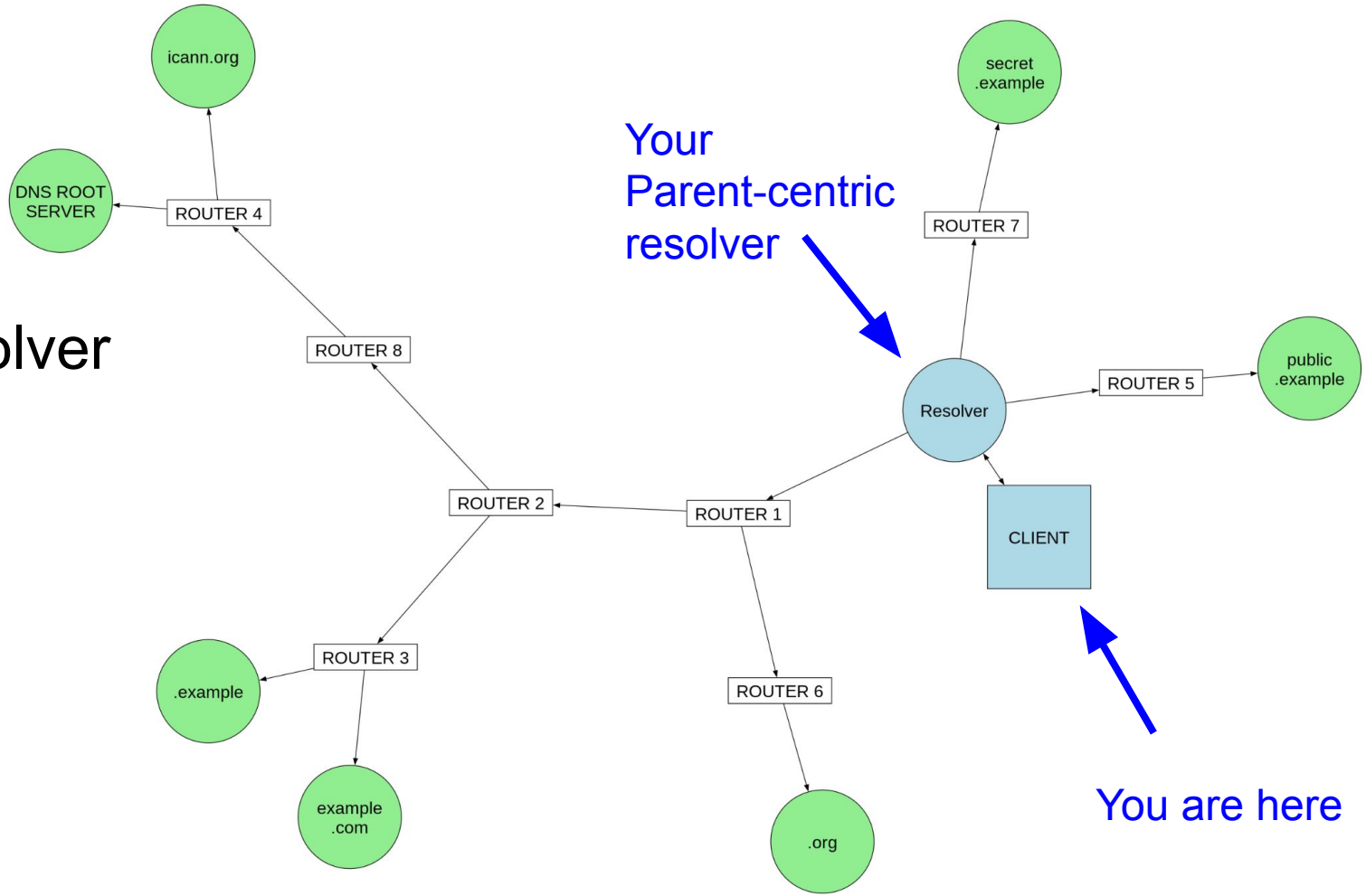
Subverting Parent-Centric DNSSEC Validating Resolvers

Wes Hardaker <ietf@hardakers.net>

Imagine Your Network's DNS Resolver

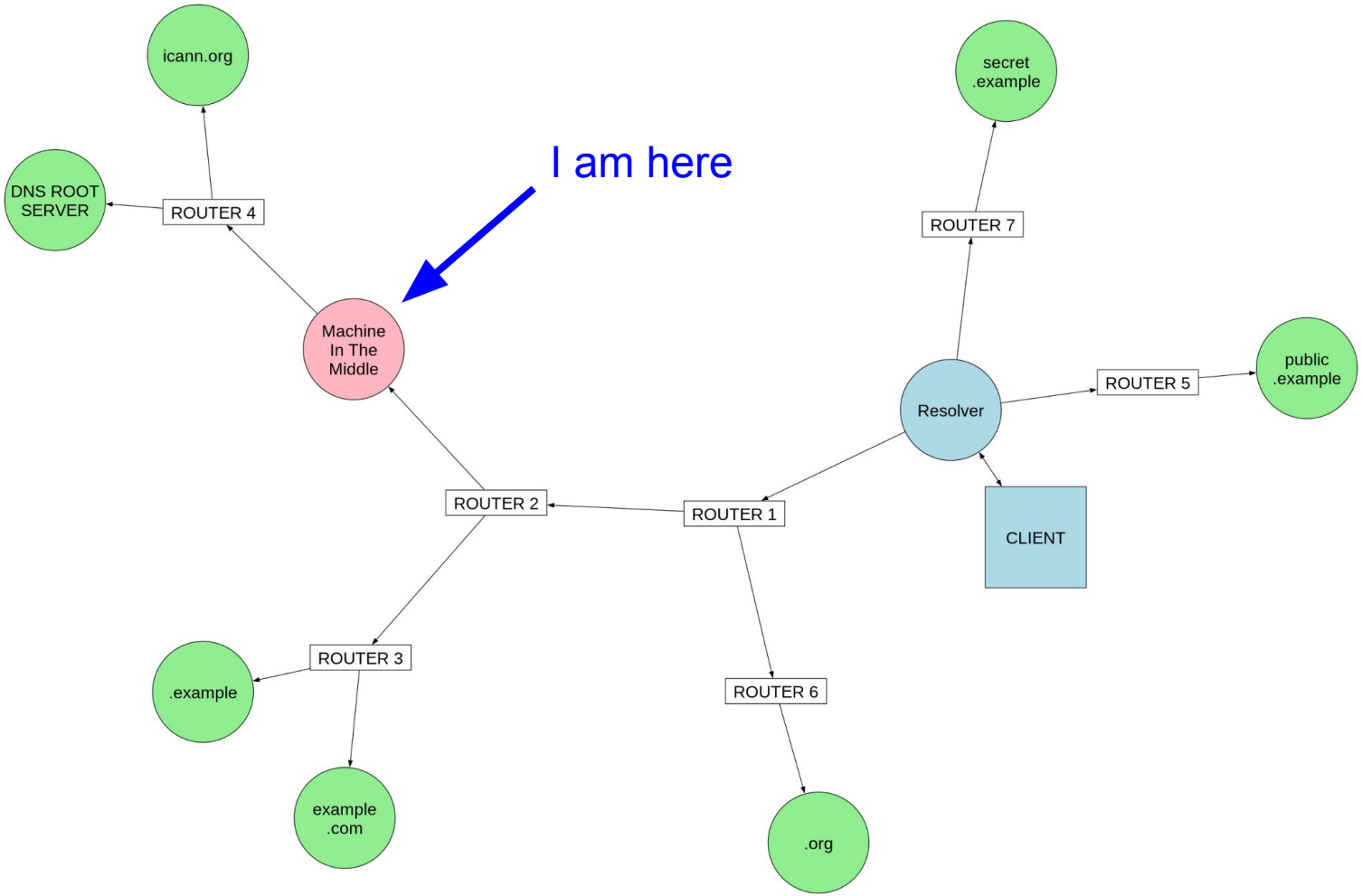
and

The Internet



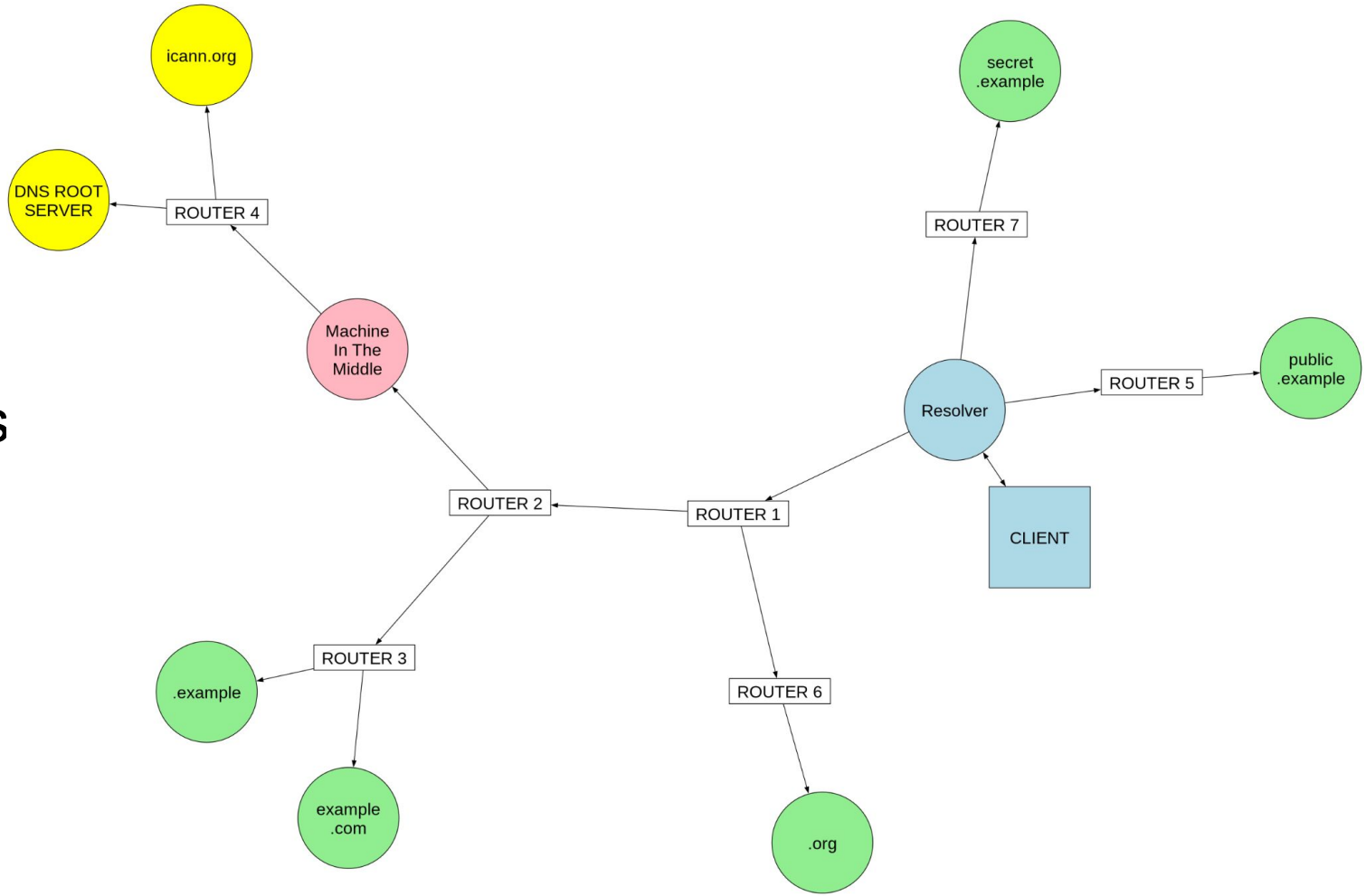
Imagine I'm In Your Network

… and I'm not a nice person



I
Can
View
All Your
Resolver's
DNS
Requests

To These
Zones

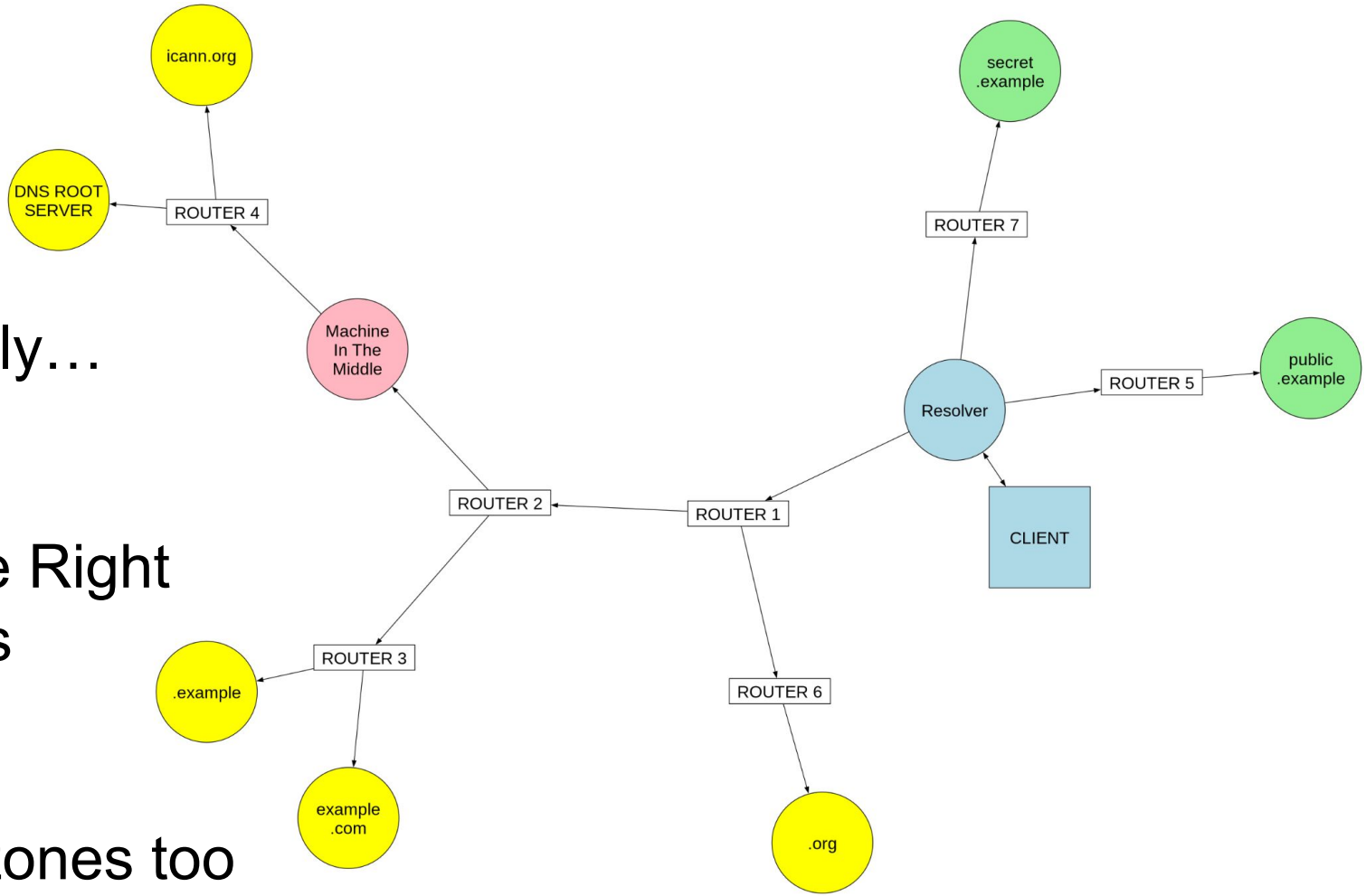


And

Surprisingly…

Under The Right Conditions

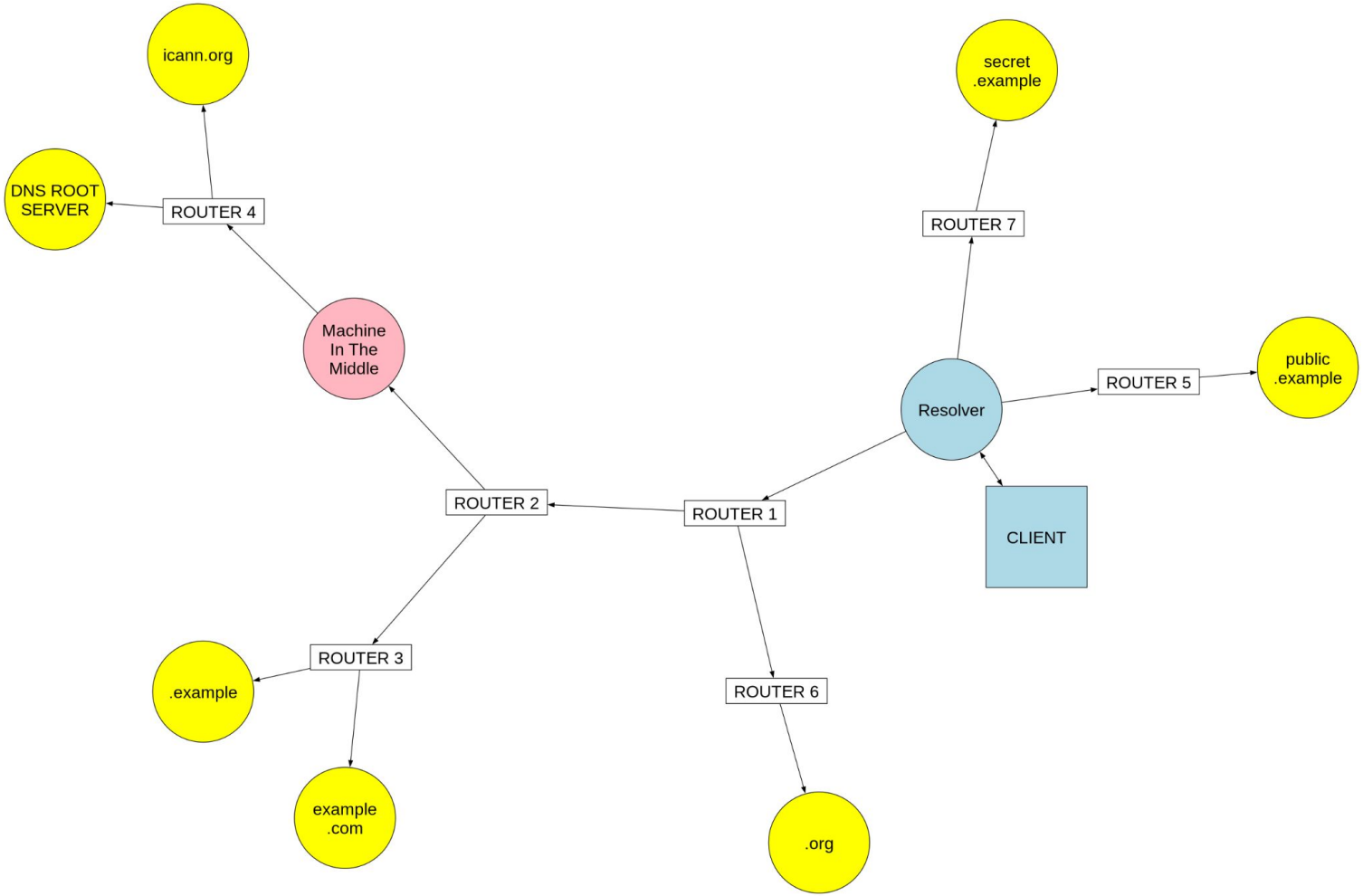
To these zones too



And

Even

These



Overview

1. Core Issue: **Parent-centric resolvers trust unvalidated data**
2. Attack demonstration
3. So what? What's the worst that could happen…
4. Mitigations

Child-centric vs Parent-centric Resolvers

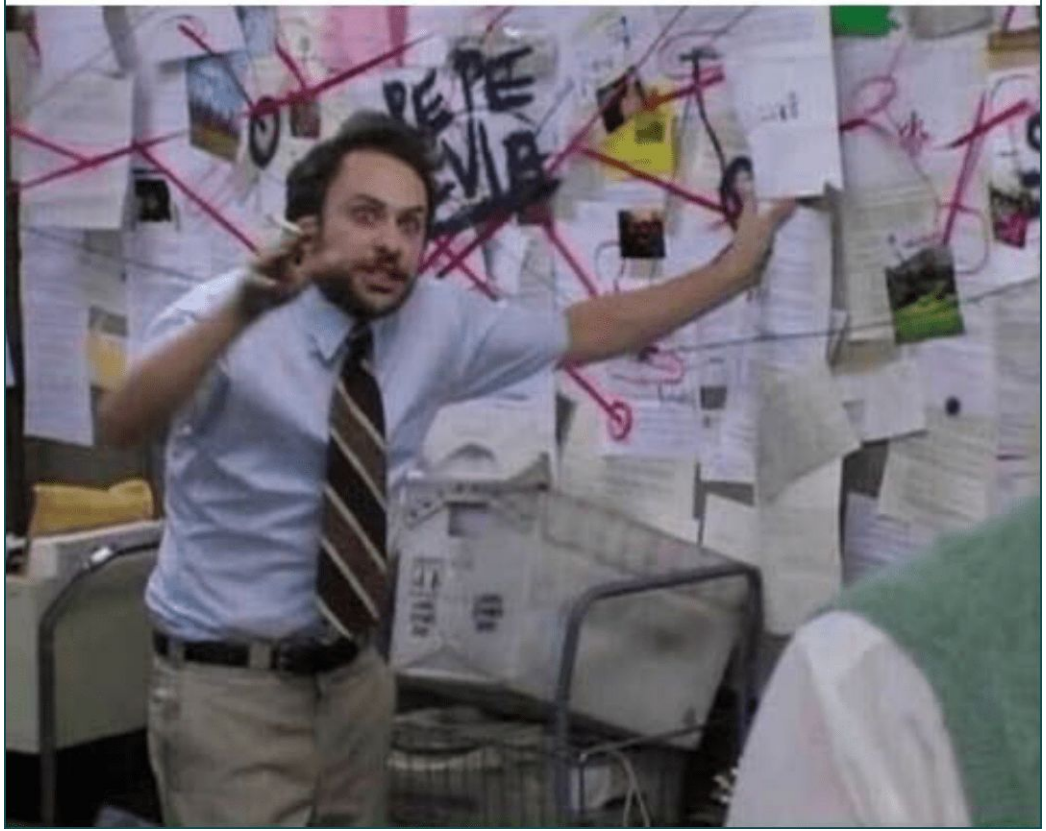
Child-centric resolvers

- Has only **one cache**: stores all data
- **DNSSEC validates** all DNS data
- **All** data is DNSSEC validated
- **Slower**

Parent-centric resolvers

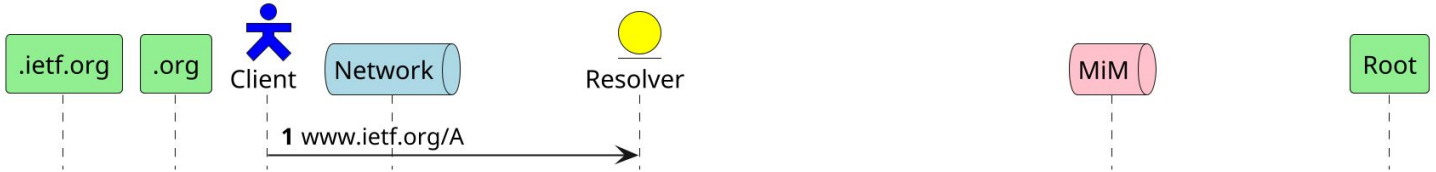
- Has **two caches**: delegation and child
- **Trusts** the parent's delegation information
- **Only client responses** are validated
- **Faster** and more "broken DNS" resilient

"what's the worst thing that could happen?"

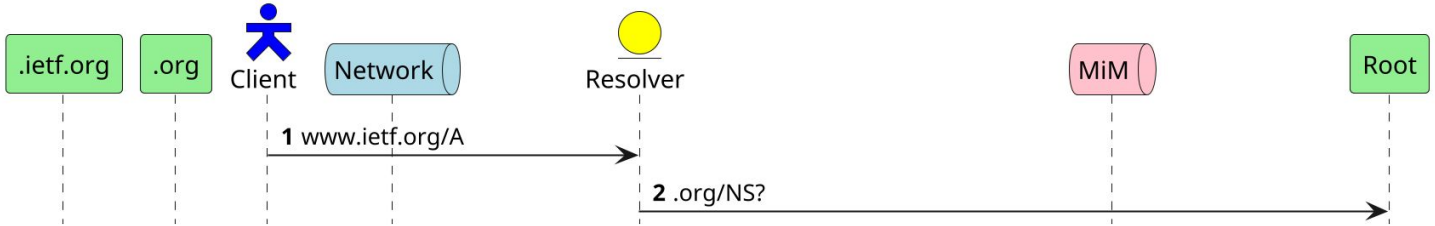



Disney
the PARENT TRAP
© 2019 Disney

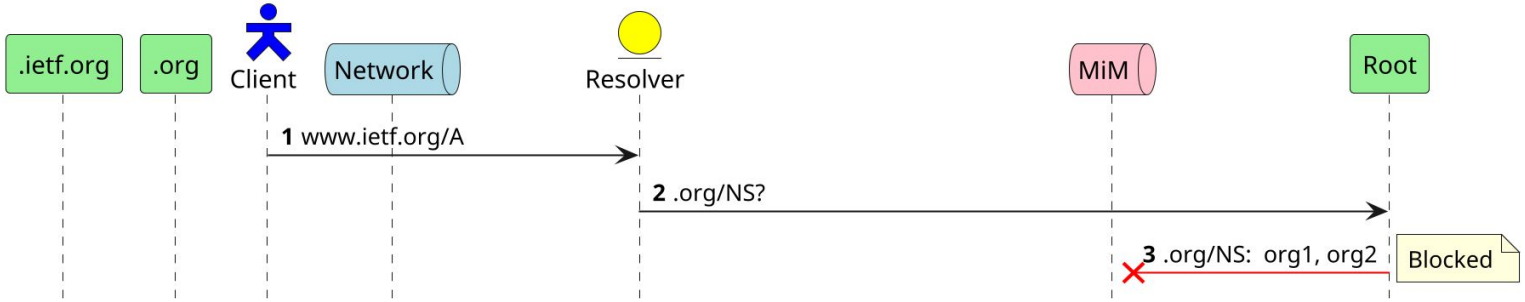
The Parent Trap



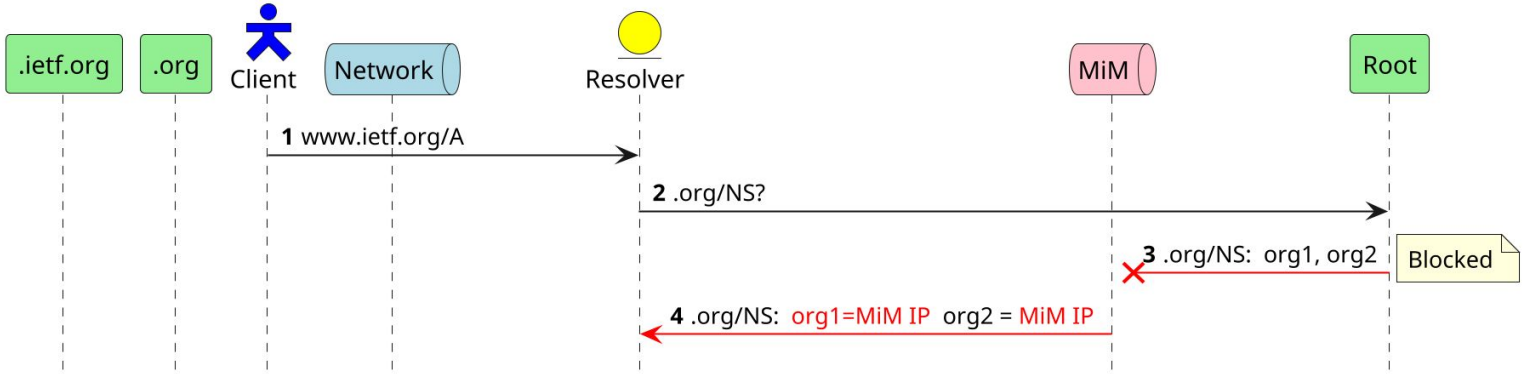
The Parent Trap



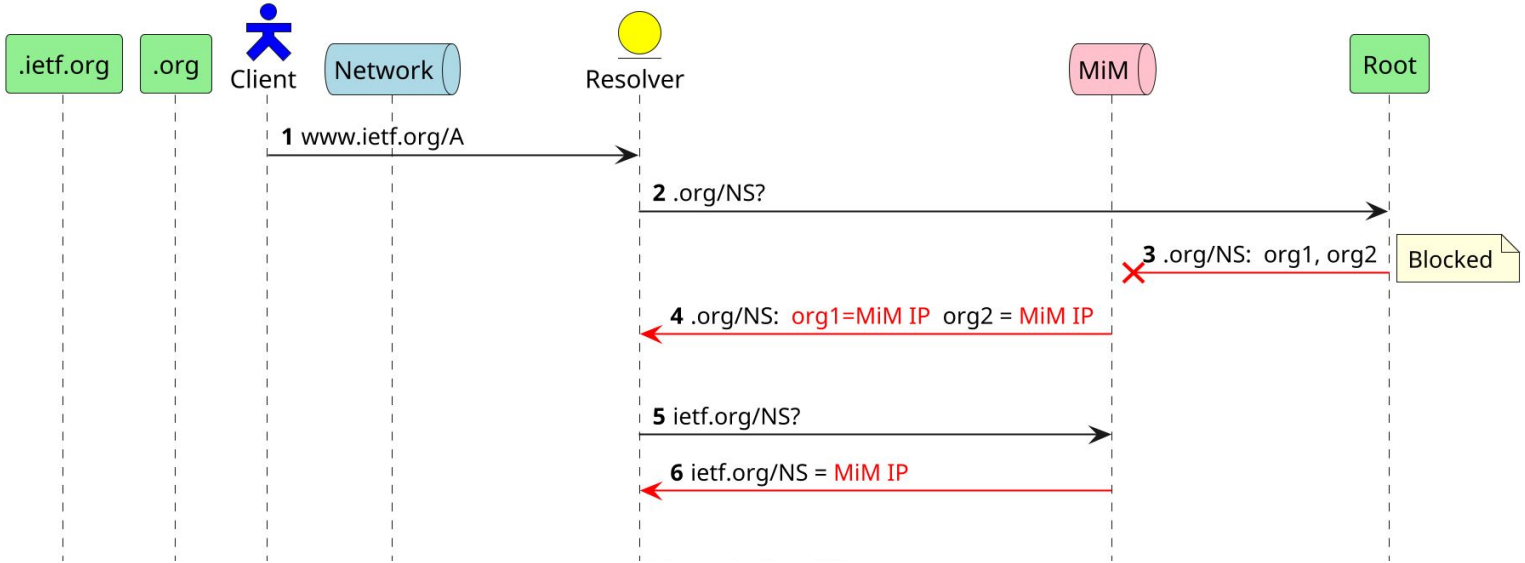
The Parent Trap



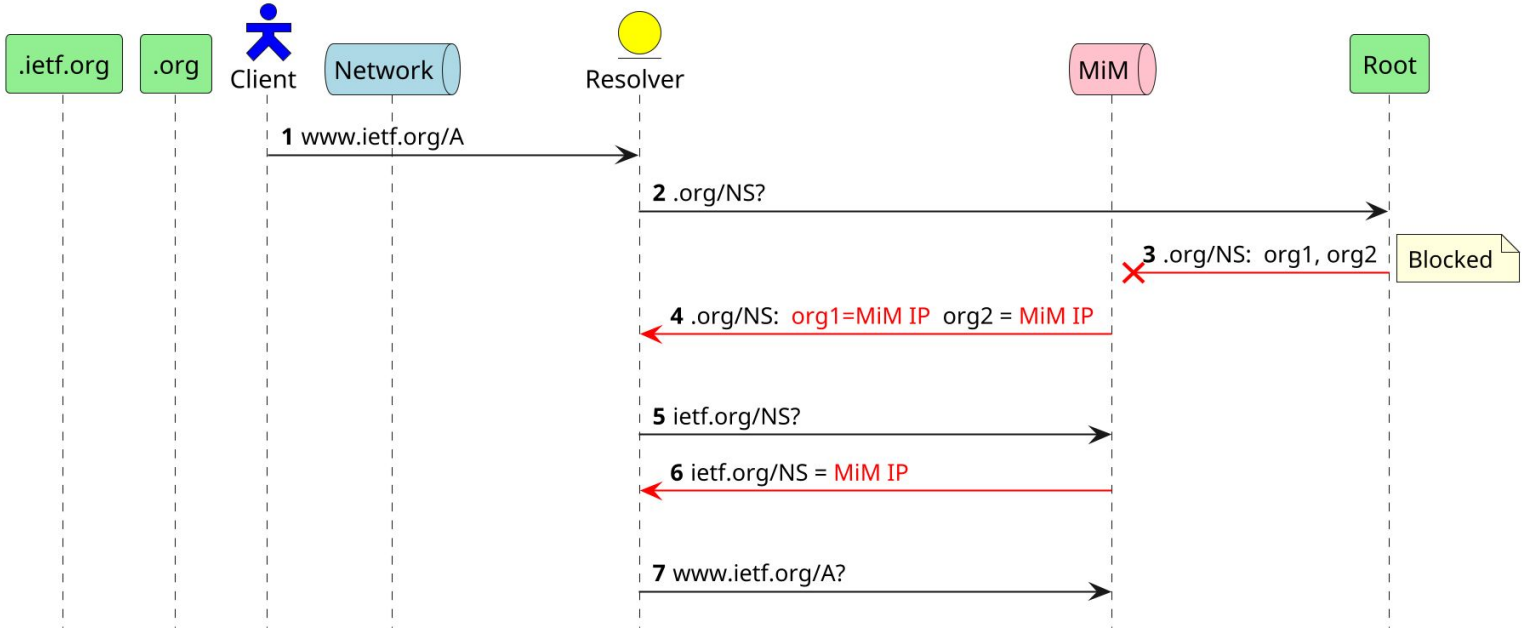
The Parent Trap



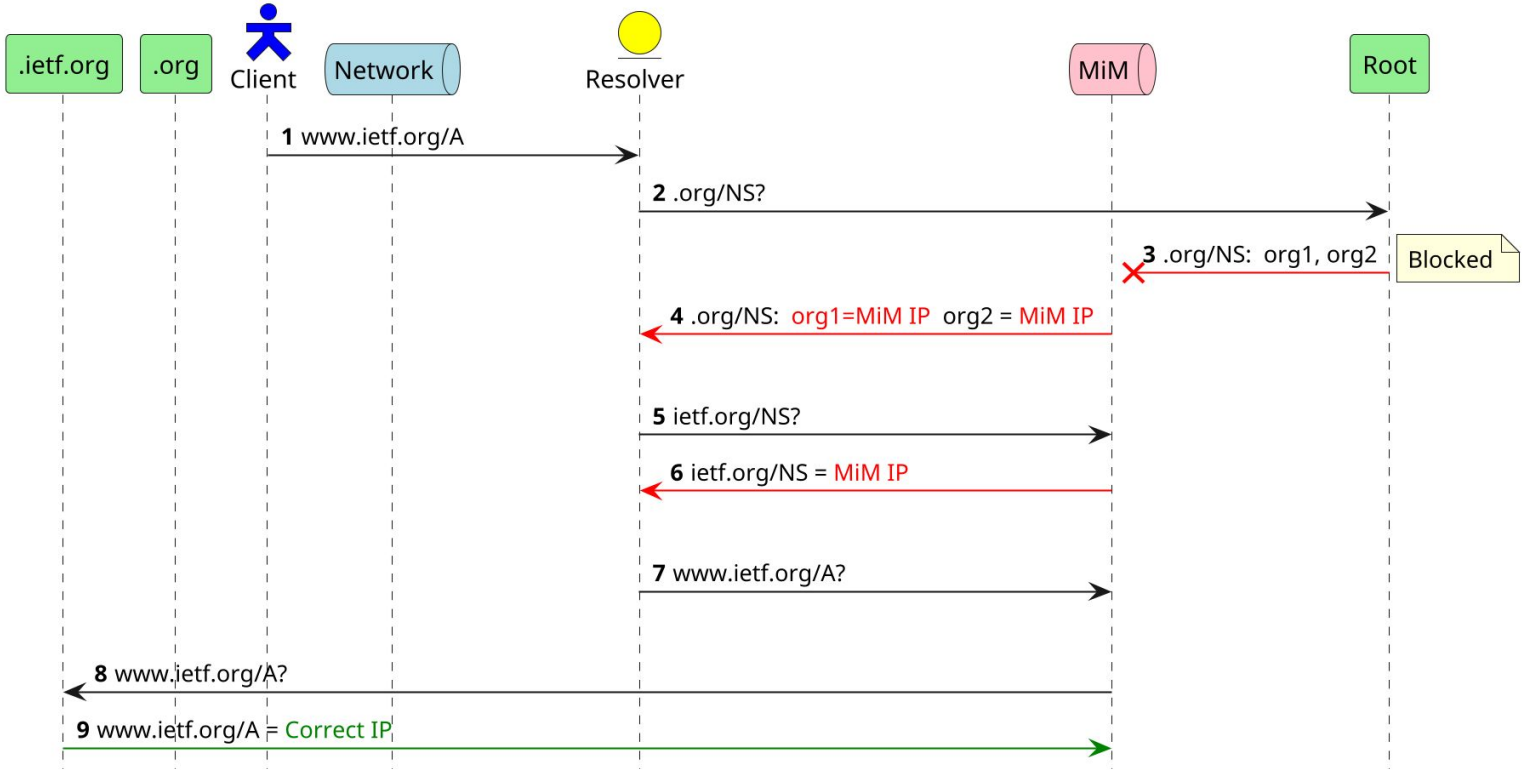
The Parent Trap



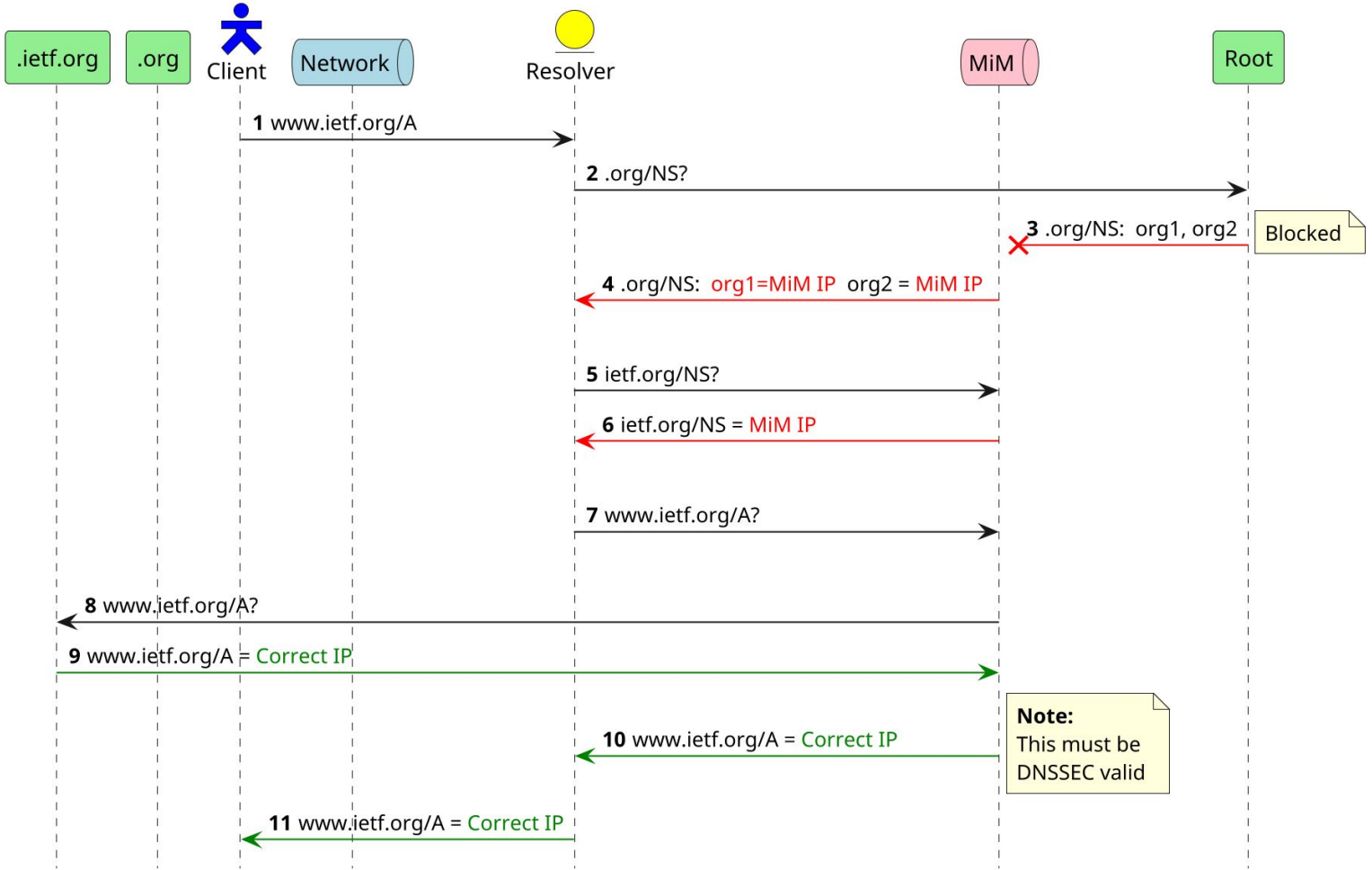
The Parent Trap

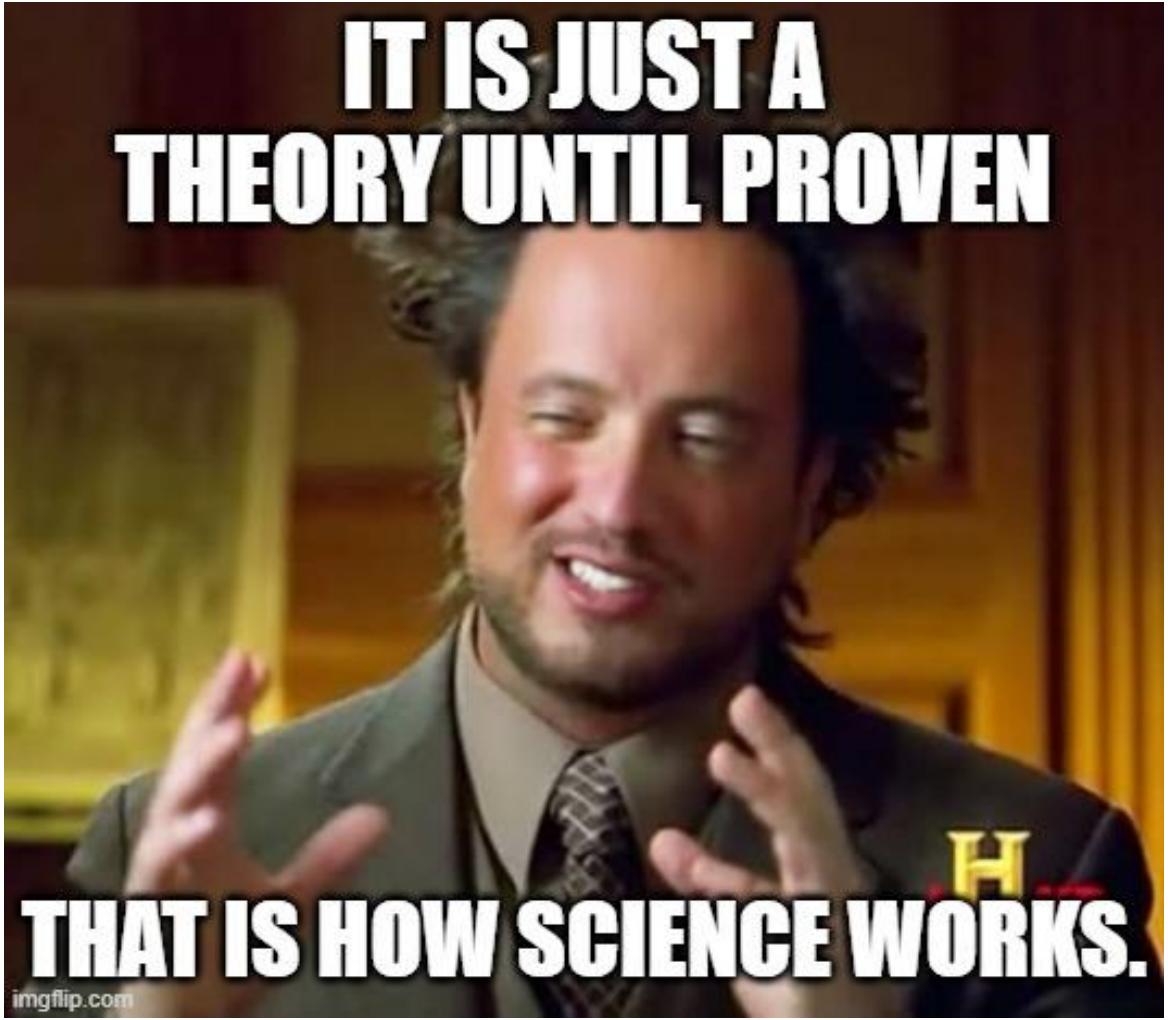


The Parent Trap



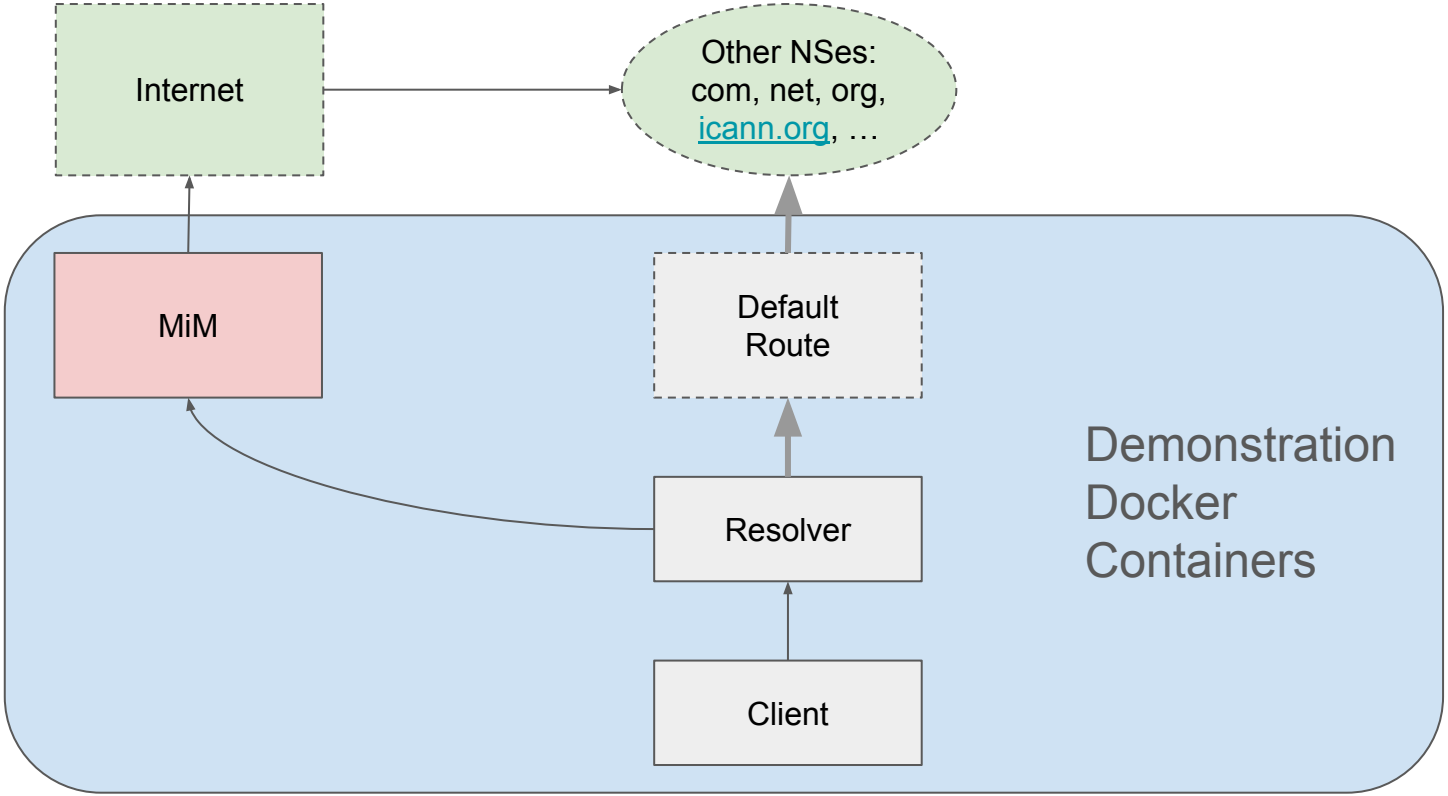
The Parent Trap



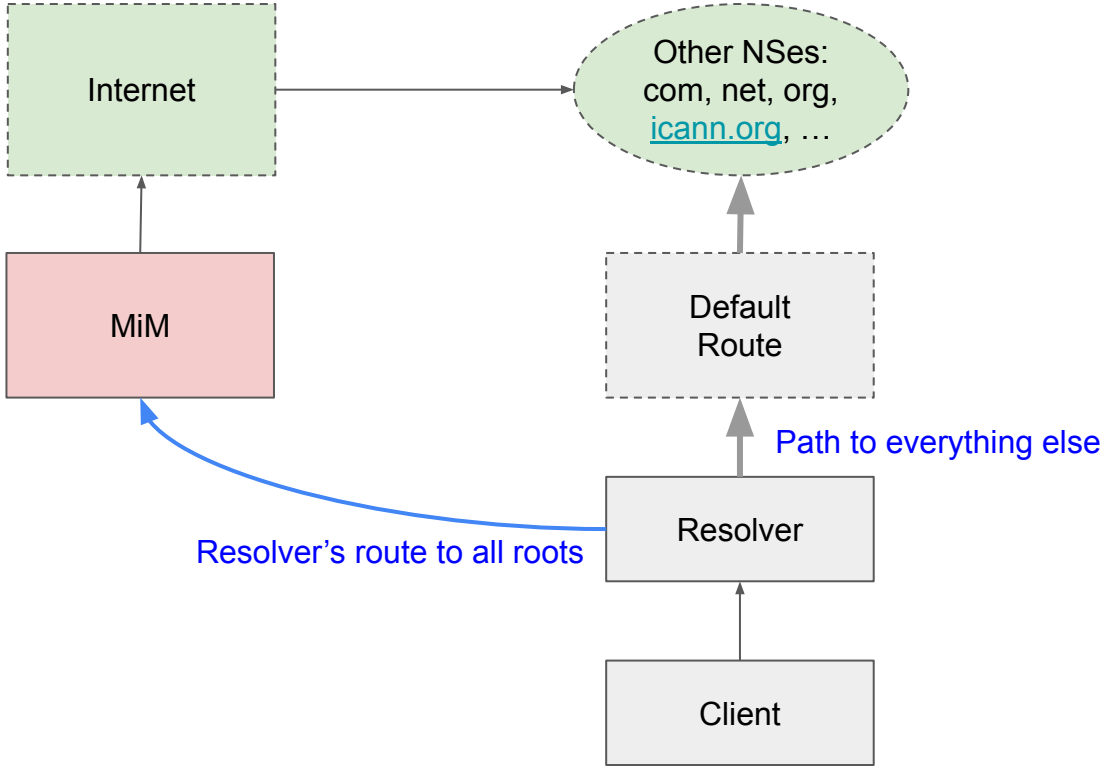

IT IS JUST A
THEORY UNTIL PROVEN
THAT IS HOW SCIENCE WORKS.
imgflip.com

Demonstration Setup

Test Network Design



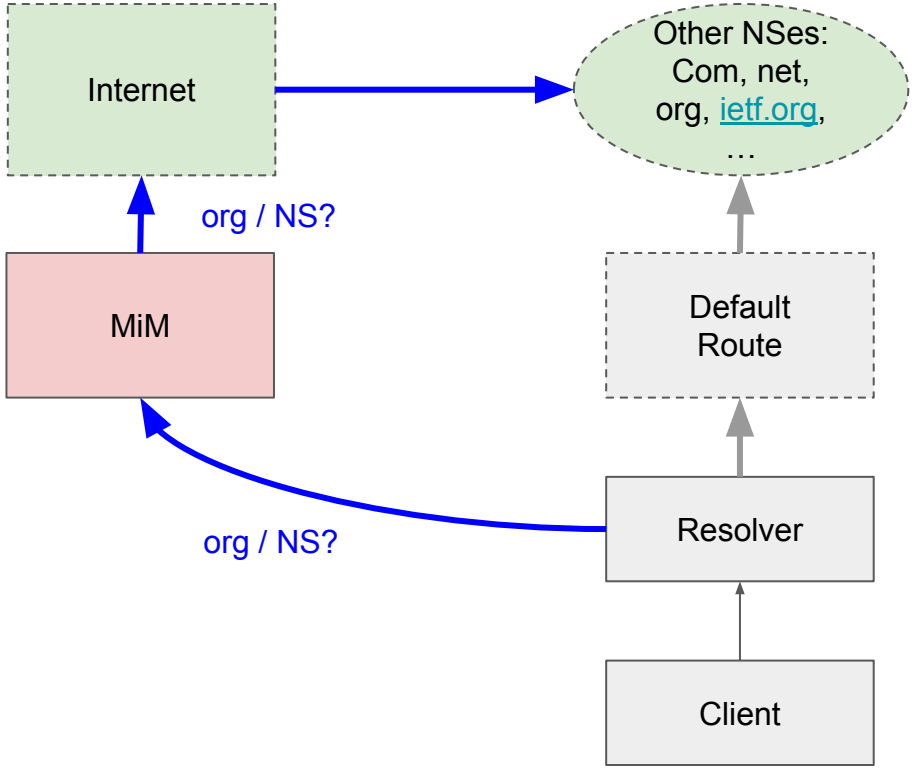
resolver to the root vs *resolver to everything else*



Parent Cache

Child Cache

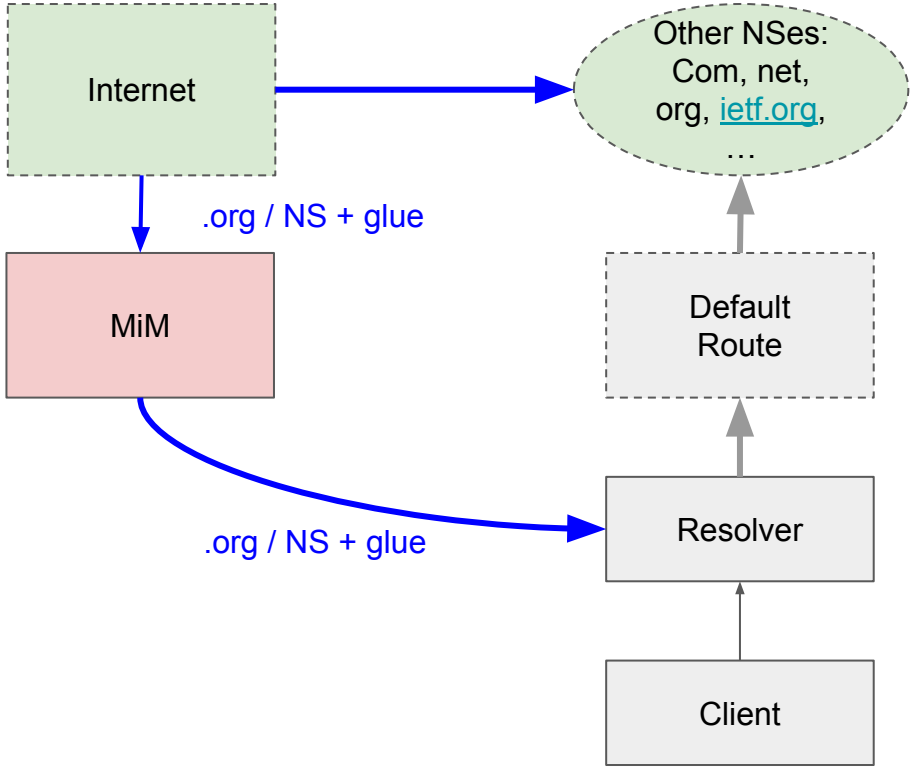
Demonstration Flow: resolver needs to know where .org is



Parent Cache

Child Cache

The DNS root will answer

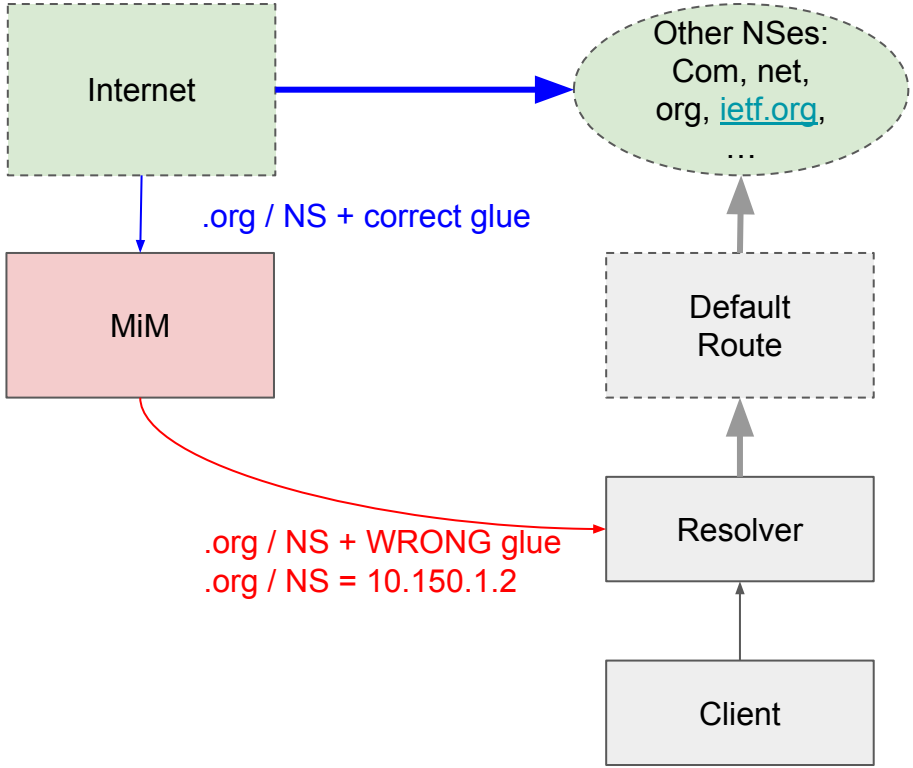


Parent Cache

.org NS Glue

Child Cache

A MiM can modify the resultts

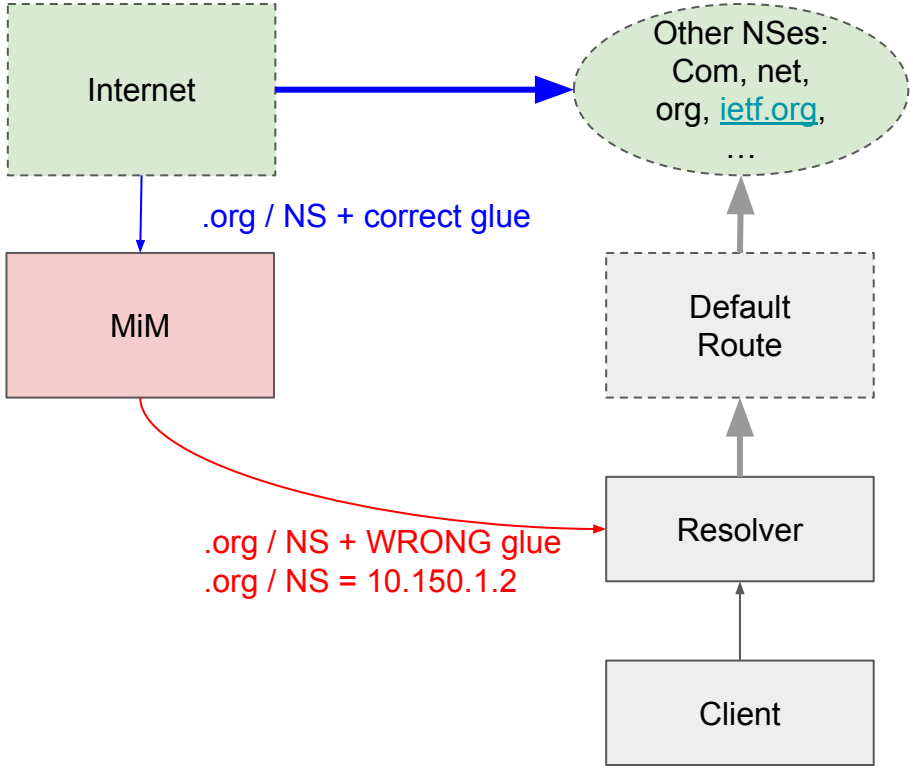


Parent Cache

.org NS Glue

Child Cache

A MiM can modify the resultts



Client-centric resolvers won't accept
Parent-centric resolvers blindly trust

Parent Cache

.org NS Glue

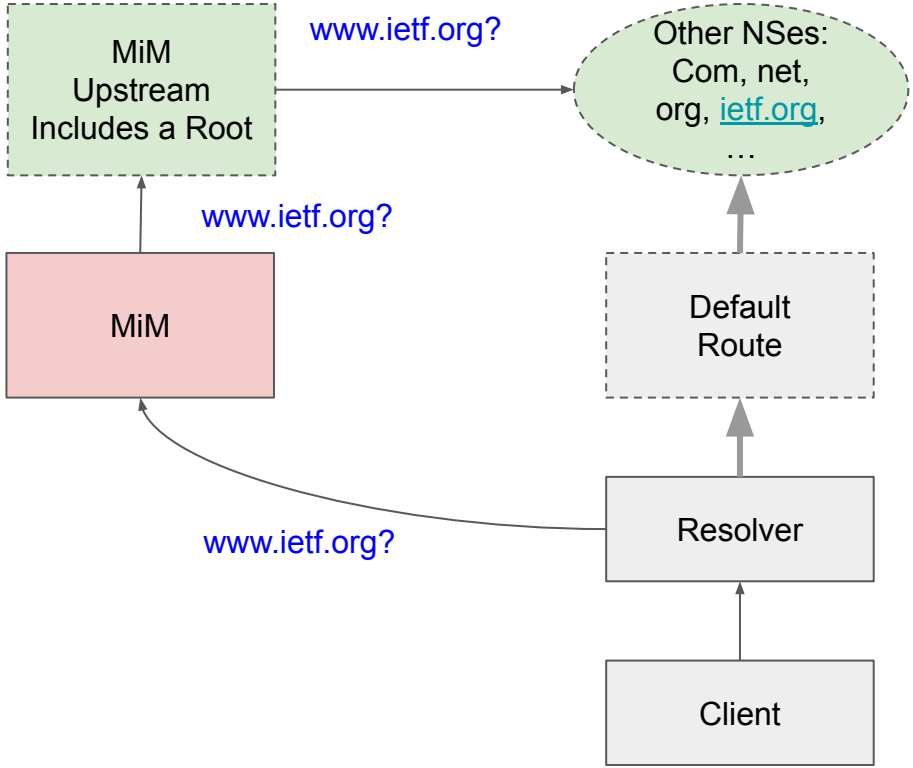
Child Cache

This continues…

- First: .org: where are you???

- Next: .ietf.org: where are you???

Demo – many steps later

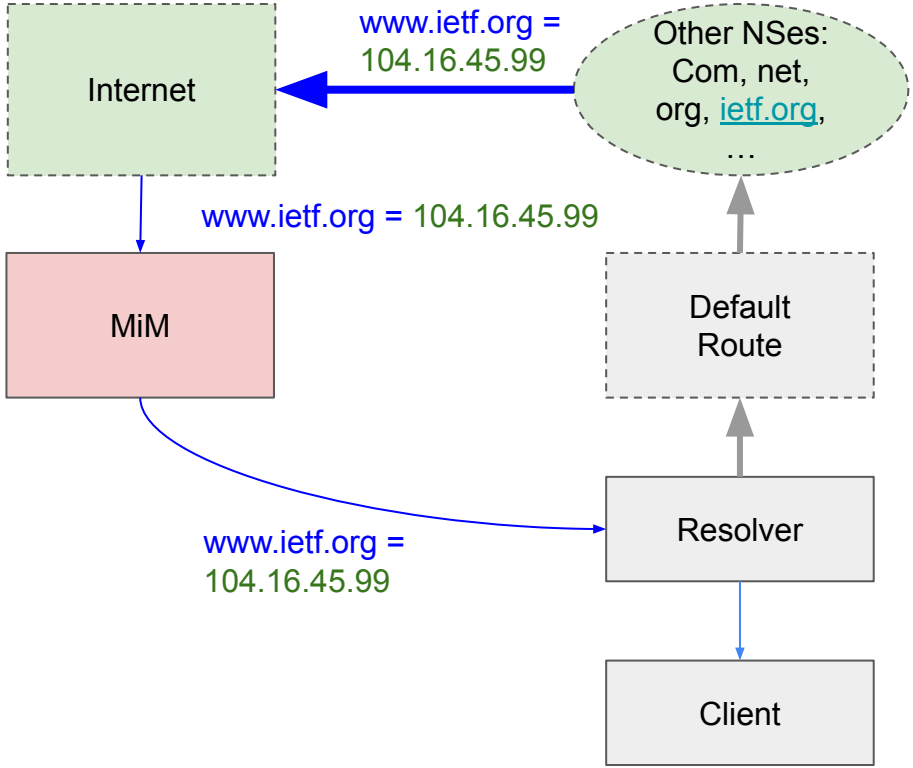


Parent Cache

.org	NS	Glue
.ietf.org	NS	Glue

Child Cache

Demo – many steps later



Parent Cache

.org	NS	Glue
.ietf.org	NS	Glue

Child Cache

www.ietf.org	A

Demonstration 1:
How a parent-centric MiM works

```
parent-centric> dig @170.247.170.2 . soa
```

```
; <<>> DiG 9.21.22-dev <<>> @170.247.170.2 . soa
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 35118
;; flags: qr aa rd; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1
;; WARNING: recursion requested but not available

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 1232
;; QUESTION SECTION:
;.                              IN      SOA

;; ANSWER SECTION:
.                       86400   IN      SOA     a.root-servers.net. nstld.verisign-grs.com. 2026110
400 1800 900 604800 86400

;; Query time: 52 msec
;; SERVER: 170.247.170.2#53(170.247.170.2) (UDP)
;; WHEN: Wed Nov 04 11:46:54 UTC 2026
;; MSG SIZE  rcvd: 103

parent-centric>
```

```
parent-centric> dig @170.247.170.2 com ns
```

```
f.gtld-servers.net.     172800  IN      A       10.150.1.2
f.gtld-servers.net.     172800  IN      A       10.150.1.2
g.gtld-servers.net.     172800  IN      A       10.150.1.2
g.gtld-servers.net.     172800  IN      A       10.150.1.2
h.gtld-servers.net.     172800  IN      A       10.150.1.2
h.gtld-servers.net.     172800  IN      A       10.150.1.2
i.gtld-servers.net.     172800  IN      A       10.150.1.2
i.gtld-servers.net.     172800  IN      A       10.150.1.2
j.gtld-servers.net.     172800  IN      A       10.150.1.2
j.gtld-servers.net.     172800  IN      A       10.150.1.2
k.gtld-servers.net.     172800  IN      A       10.150.1.2
k.gtld-servers.net.     172800  IN      A       10.150.1.2
l.gtld-servers.net.     172800  IN      A       10.150.1.2
l.gtld-servers.net.     172800  IN      A       10.150.1.2
m.gtld-servers.net.     172800  IN      A       10.150.1.2
m.gtld-servers.net.     172800  IN      A       10.150.1.2

;; Query time: 137 msec
;; SERVER: 170.247.170.2#53(170.247.170.2) (UDP)
;; WHEN: Wed Nov 04 11:47:21 UTC 2026
;; MSG SIZE  rcvd: 1618

parent-centric>
```

```
parent-centric> dig @10.150.1.2 google.com ns
```

```
google.com.             172800  IN      NS      ns3.google.com.
google.com.             172800  IN      NS      ns4.google.com.

;; ADDITIONAL SECTION:
google.com.             172800  IN      A       10.150.1.3
google.com.             172800  IN      A       10.150.1.3
google.com.             172800  IN      A       10.150.1.3
google.com.             172800  IN      A       10.150.1.3
ns2.google.com.         172800  IN      A       10.150.1.3
ns2.google.com.         172800  IN      A       10.150.1.3
ns1.google.com.         172800  IN      A       10.150.1.3
ns1.google.com.         172800  IN      A       10.150.1.3
ns3.google.com.         172800  IN      A       10.150.1.3
ns3.google.com.         172800  IN      A       10.150.1.3
ns4.google.com.         172800  IN      A       10.150.1.3
ns4.google.com.         172800  IN      A       10.150.1.3

;; Query time: 68 msec
;; SERVER: 10.150.1.2#53(10.150.1.2) (UDP)
;; WHEN: Wed Nov 04 11:48:15 UTC 2026
;; MSG SIZE  rcvd: 535

parent-centric>
```

```
google.com.             172800  IN      NS      ns3.google.com.
google.com.             172800  IN      NS      ns4.google.com.

;; ADDITIONAL SECTION:
google.com.             172800  IN      A       10.150.1.3
google.com.             172800  IN      A       10.150.1.3
google.com.             172800  IN      A       10.150.1.3
google.com.             172800  IN      A       10.150.1.3
ns2.google.com.         172800  IN      A       10.150.1.3
ns2.google.com.         172800  IN      A       10.150.1.3
ns1.google.com.         172800  IN      A       10.150.1.3
ns1.google.com.         172800  IN      A       10.150.1.3
ns3.google.com.         172800  IN      A       10.150.1.3
ns3.google.com.         172800  IN      A       10.150.1.3
ns4.google.com.         172800  IN      A       10.150.1.3
ns4.google.com.         172800  IN      A       10.150.1.3

;; Query time: 68 msec
;; SERVER: 10.150.1.2#53(10.150.1.2) (UDP)
;; WHEN: Wed Nov 04 11:48:15 UTC 2026
;; MSG SIZE  rcvd: 535

parent-centric> dig @10.150.1.3 www.google.com
```

```
parent-centric> dig @10.150.1.3 www.google.com
```

```
;; WARNING: recursion requested but not available

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 512
;; QUESTION SECTION:
;www.google.com.                        IN      A

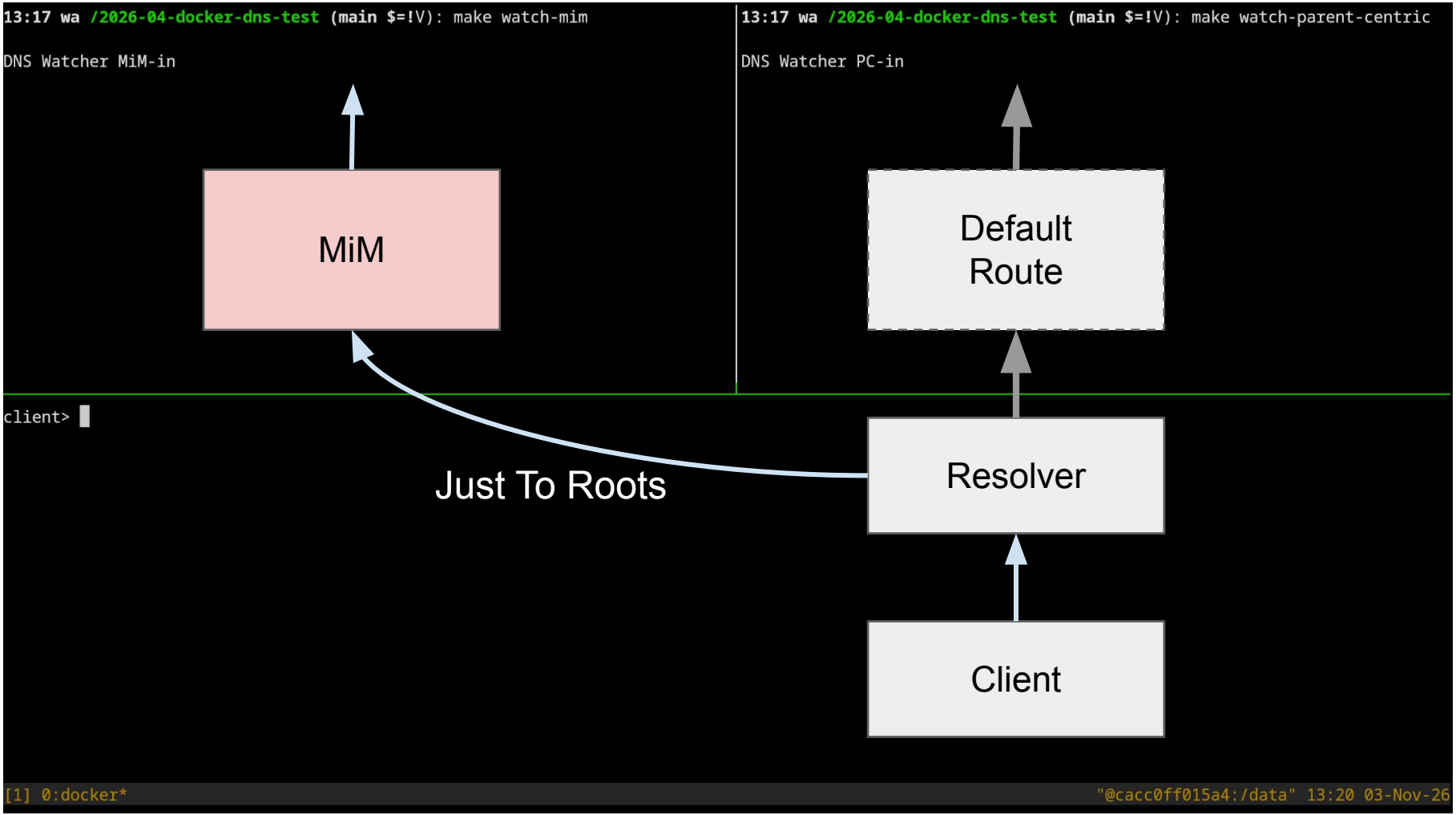
;; ANSWER SECTION:
www.google.com.         300     IN      A       6.6.6.6
www.google.com.         300     IN      A       6.6.6.6
www.google.com.         300     IN      A       6.6.6.6
www.google.com.         300     IN      A       6.6.6.6
www.google.com.         300     IN      A       6.6.6.6
www.google.com.         300     IN      A       6.6.6.6
www.google.com.         300     IN      A       6.6.6.6
www.google.com.         300     IN      A       6.6.6.6

;; Query time: 75 msec
;; SERVER: 10.150.1.3#53(10.150.1.3) (UDP)
;; WHEN: Wed Nov 04 11:50:04 UTC 2026
;; MSG SIZE  rcvd: 283

parent-centric>
```

Demonstration 2:
No MiM – Just Observation


13:17 wa /2026-04-docker-dns-test (main $=!V): make watch-mim
DNS Watcher MiM-in
13:17 wa /2026-04-docker-dns-test (main $=!V): make watch-parent-centric
DNS Watcher PC-in
client>
[1] 0:docker*
"@cacc0ff015a4:/data" 13:20 03-Nov-26


13:17 wa /2026-04-docker-dns-test (main $=!V): make watch-mim
DNS Watcher MiM-in
13:17 wa /2026-04-docker-dns-test (main $=!V): make watch-parent-centric
DNS Watcher PC-in
MiM
Default Route
client>
Just To Roots
Resolver
Client
[1] 0:docker*
"@cacc0ff015a4:/data" 13:20 03-Nov-26


13:17 wa /2026-04-docker-dns-test (main $=!V): make watch-mim
DNS Watcher MiM-in
13:17 wa /2026-04-docker-dns-test (main $=!V): make watch-parent-centric
DNS Watcher PC-in
client> dig @parent-centric google.com
[1] 0:docker*
"@cacc0ff015a4:/data" 13:20 03-Nov-26

```
13:17 wa /2026-04-docker-dns-test (main $=!V): make watch-mim

DNS Watcher MiM-in

MiM-in  >  NS .
MiM-in  >  NS com.
MiM-in  >  DNSKEY .
```

```
13:17 wa /2026-04-docker-dns-test (main $=!V): make watch-parent-centric

DNS Watcher PC-in

PC-in   >  A google.com.
PC-in   >  NS com.
PC-in   >  DNSKEY com.
PC-in   >  DS google.com.
```

```
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

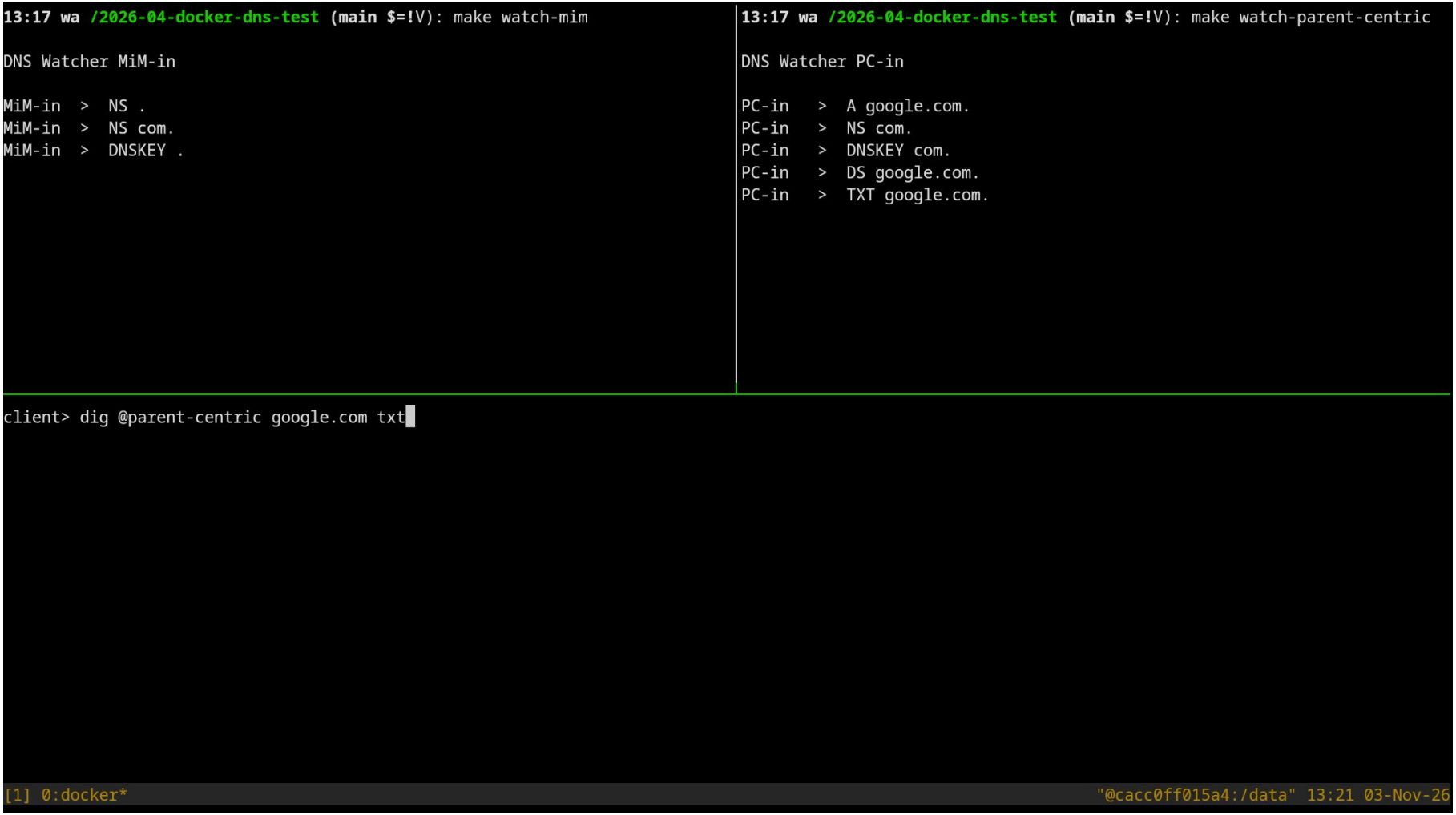
;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: 9c591c608044d1a7010000006aea5034d7c8a7748244b5d7 (good)
;; QUESTION SECTION:
;google.com.                    IN      A

;; ANSWER SECTION:
google.com.             300     IN      A       142.250.217.142

;; Query time: 830 msec
;; SERVER: 10.200.0.3#53(parent-centric) (UDP)
;; WHEN: Tue Nov 03 21:20:52 UTC 2026
;; MSG SIZE  rcvd: 83

client>
```

```
[1] 0:docker*                                   "@cacc0ff015a4:/data" 13:20 03-Nov-26
```


13:17 wa /2026-04-docker-dns-test (main $=!V): make watch-mim
DNS Watcher MiM-in
MiM-in > NS .
MiM-in > NS com.
MiM-in > DNSKEY .
13:17 wa /2026-04-docker-dns-test (main $=!V): make watch-parent-centric
DNS Watcher PC-in
PC-in > A google.com.
PC-in > NS com.
PC-in > DNSKEY com.
PC-in > DS google.com.
PC-in > TXT google.com.
client> dig @parent-centric google.com txt
[1] 0:docker*
"@cacc0ff015a4:/data" 13:21 03-Nov-26

```
13:17 wa /2026-04-docker-dns-test (main $=!V): make watch-mim

DNS Watcher MiM-in

MiM-in  >  NS .
MiM-in  >  NS com.
MiM-in  >  DNSKEY .
```

```
13:17 wa /2026-04-docker-dns-test (main $=!V): make watch-parent-centric

DNS Watcher PC-in

PC-in   >  A google.com.
PC-in   >  NS com.
PC-in   >  DNSKEY com.
PC-in   >  DS google.com.
PC-in   >  TXT google.com.
```

```
google.com.             286     IN      TXT     "v=spf1 include:_spf.google.com ~all"
google.com.             286     IN      TXT     "apple-domain-verification=30afIBcvSuDV2PLX"
google.com.             286     IN      TXT     "MS=E4A68B9AB2BB9670BCE15412F62916164C0B20BB"
google.com.             286     IN      TXT     "docusign=05958488-4752-4ef2-95eb-aa7ba8a3bd0e"
google.com.             286     IN      TXT     "docusign=1b0a6754-49b1-4db5-8540-d2c12664b289"
google.com.             286     IN      TXT     "facebook-domain-verification=22rm551cu4k0ab0bxsw536tlds4h95"
google.com.             286     IN      TXT     "onetrust-domain-verification=0d477fe608074e6f9c12bca7826035cc"
google.com.             286     IN      TXT     "onetrust-domain-verification=6d685f1d41a94696ad7ef771f68993e0"
google.com.             286     IN      TXT     "globalsign-smime-dv=CDYX+XFHUw2wml6/Gb8+59BsH31KzUr6c1l2BPvqKX8="
google.com.             286     IN      TXT     "work-accounts-domain-verification=Tcj6JjIMZOw2KsSEw2Nt2rLae89tN6"

;; Query time: 0 msec
;; SERVER: 10.200.0.3#53(parent-centric) (UDP)
;; WHEN: Tue Nov 03 21:21:49 UTC 2026
;; MSG SIZE  rcvd: 1215

client>
```

```
[1] 0:docker*                                                    "@cacc0ff015a4:/data" 13:21 03-Nov-26
```

```
13:17 wa /2026-04-docker-dns-test (main $=!V): make watch-mim

DNS Watcher MiM-in

MiM-in  >  NS .
MiM-in  >  NS com.
MiM-in  >  DNSKEY .
```

```
13:17 wa /2026-04-docker-dns-test (main $=!V): make watch-parent-centric

DNS Watcher PC-in

PC-in   >  A google.com.
PC-in   >  NS com.
PC-in   >  DNSKEY com.
PC-in   >  DS google.com.
PC-in   >  TXT google.com.
```

```
client> dig @parent-centric google.com _dmarc.google.com
```

```
[1] 0:docker*                                "@cacc0ff015a4:/data" 13:23 03-Nov-26
```

```
13:17 wa /2026-04-docker-dns-test (main $=!V): make watch-mim

DNS Watcher MiM-in

MiM-in  >  NS .
MiM-in  >  NS com.
MiM-in  >  DNSKEY .
```

```
13:17 wa /2026-04-docker-dns-test (main $=!V): make watch-parent-centric

DNS Watcher PC-in

PC-in   >  A google.com.
PC-in   >  NS com.
PC-in   >  DNSKEY com.
PC-in   >  DS google.com.
PC-in   >  TXT google.com.
PC-in   >  A _dmarc.google.com.
PC-in   >  TXT _dmarc.google.com.
```

```
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: 16154e6d13899888010000006aea50f3d02985805781352e (good)
;; QUESTION SECTION:
;_dmarc.google.com.             IN      TXT

;; ANSWER SECTION:
_dmarc.google.com.      300     IN      TXT     "v=DMARC1; p=reject; rua=mailto:mailauth-reports@google.com"

;; Query time: 21 msec
;; SERVER: 10.200.0.3#53(parent-centric) (UDP)
;; WHEN: Tue Nov 03 21:24:03 UTC 2026
;; MSG SIZE  rcvd: 145

client>
```

```
[1] 0:docker*                                                    "@cacc0ff015a4:/data" 13:24 03-Nov-26
```

Demonstration 3:
Attacking a parent-centric resolver


13:25 wa /2026-04-docker-dns-test (main $=!V): make watch-mim
DNS Watcher MiM-in
13:25 wa /2026-04-docker-dns-test (main $=!V): make watch-parent-centric
DNS Watcher PC-in
client> dig @parent-centric google.com
[1] 0:docker*
"@489e58591910:/data" 13:26 03-Nov-26

```
13:25 wa /2026-04-docker-dns-test (main $=!V): make watch-mim

DNS Watcher MiM-in

MiM-in  >  NS .
MiM-in  >  NS com.
MiM-in  >  DNSKEY .
MiM-in  >  DNSKEY com.
MiM-in  >  A google.com.
MiM-in  >  DS google.com.
```

```
13:25 wa /2026-04-docker-dns-test (main $=!V): make watch-parent-centric

DNS Watcher PC-in

PC-in   >  A google.com.
```

```
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: 27c82197e9db593f010000006aea518ee8eb418e995abf3f (good)
;; QUESTION SECTION:
;google.com.                    IN      A

;; ANSWER SECTION:
google.com.             300     IN      A       142.250.217.142

;; Query time: 523 msec
;; SERVER: 10.200.0.3#53(parent-centric) (UDP)
;; WHEN: Tue Nov 03 21:26:38 UTC 2026
;; MSG SIZE  rcvd: 83

client>
```

```
[1] 0:docker*                                                    "@489e58591910:/data" 13:26 03-Nov-26
```

```
13:25 wa /2026-04-docker-dns-test (main $=!V): make watch-mim

DNS Watcher MiM-in

MiM-in  >  NS .
MiM-in  >  NS com.
MiM-in  >  DNSKEY .
MiM-in  >  DNSKEY com.
MiM-in  >  A google.com.
MiM-in  >  DS google.com.
```

```
13:25 wa /2026-04-docker-dns-test (main $=!V): make watch-parent-centric

DNS Watcher PC-in

PC-in   >  A google.com.
```

```
client> dig @parent-centric google.com txt
```

```
[1] 0:docker*                                "@489e58591910:/data" 13:27 03-Nov-26
```

```
13:25 wa /2026-04-docker-dns-test (main $=!V): make watch-mim

DNS Watcher MiM-in

MiM-in  >  NS .
MiM-in  >  NS com.
MiM-in  >  DNSKEY .
MiM-in  >  DNSKEY com.
MiM-in  >  A google.com.
MiM-in  >  DS google.com.
MiM-in  >  TXT google.com.
```

```
13:25 wa /2026-04-docker-dns-test (main $=!V): make watch-parent-centric

DNS Watcher PC-in

PC-in   >  A google.com.
PC-in   >  TXT google.com.
```

```
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: fbcce1304abd1c4c010000006aea51caecd624ad8d7bdd24 (good)
;; QUESTION SECTION:
;google.com.                    IN      TXT

;; ANSWER SECTION:
google.com.             300     IN      TXT     "v=spf1 mx ip4:6.6.6.6"

;; Query time: 128 msec
;; SERVER: 10.200.0.3#53(parent-centric) (UDP)
;; WHEN: Tue Nov 03 21:27:38 UTC 2026
;; MSG SIZE  rcvd: 101

client>
```

```
[1] 0:docker*                                                    "@489e58591910:/data" 13:27 03-Nov-26
```


13:25 wa /2026-04-docker-dns-test (main $=!V): make watch-mim
DNS Watcher MiM-in
MiM-in > NS .
MiM-in > NS com.
MiM-in > DNSKEY .
MiM-in > DNSKEY com.
MiM-in > A google.com.
MiM-in > DS google.com.
MiM-in > TXT google.com.
13:25 wa /2026-04-docker-dns-test (main $=!V): make watch-parent-centric
DNS Watcher PC-in
PC-in > A google.com.
PC-in > TXT google.com.
client> dig @parent-centric google.com mx
[1] 0:docker*
"@489e58591910:/data" 13:28 03-Nov-26

```
13:25 wa /2026-04-docker-dns-test (main $=!V): make watch-mim

DNS Watcher MiM-in

MiM-in  >  NS .
MiM-in  >  NS com.
MiM-in  >  DNSKEY .
MiM-in  >  DNSKEY com.
MiM-in  >  A google.com.
MiM-in  >  DS google.com.
MiM-in  >  TXT google.com.
MiM-in  >  MX google.com.
```

```
13:25 wa /2026-04-docker-dns-test (main $=!V): make watch-parent-centric

DNS Watcher PC-in

PC-in   >  A google.com.
PC-in   >  TXT google.com.
PC-in   >  MX google.com.
```

```
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

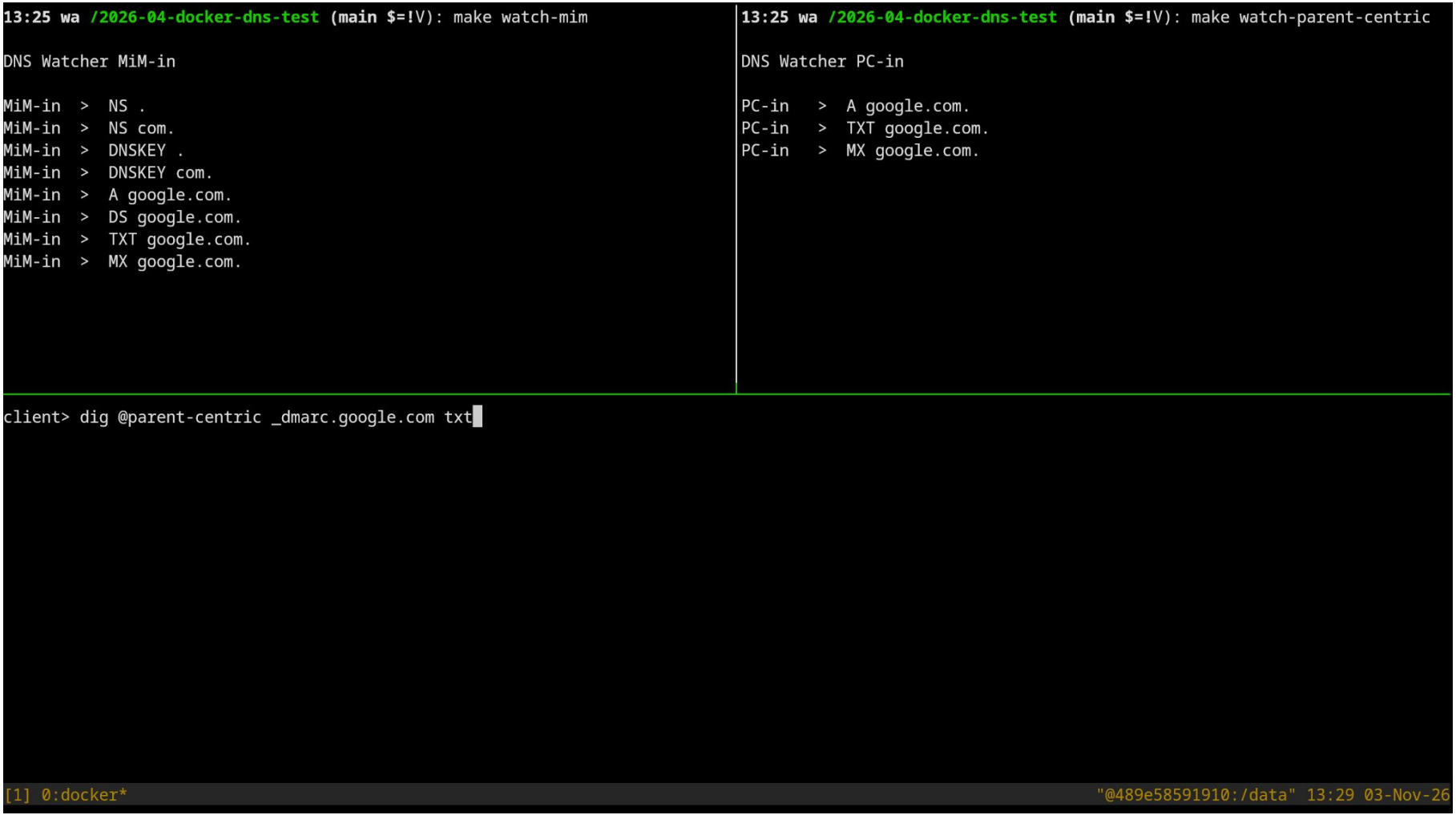
;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: c16eb47f9a42de24010000006aea521be3a56ec65d7a9d7d (good)
;; QUESTION SECTION:
;google.com.                    IN      MX

;; ANSWER SECTION:
google.com.             300     IN      MX      10 smtp.evil.example.

;; Query time: 96 msec
;; SERVER: 10.200.0.3#53(parent-centric) (UDP)
;; WHEN: Tue Nov 03 21:28:59 UTC 2026
;; MSG SIZE  rcvd: 100

client>
```

```
[1] 0:docker*                                                    "@489e58591910:/data" 13:29 03-Nov-26
```


13:25 wa /2026-04-docker-dns-test (main $=!V): make watch-mim
DNS Watcher MiM-in
MiM-in > NS .
MiM-in > NS com.
MiM-in > DNSKEY .
MiM-in > DNSKEY com.
MiM-in > A google.com.
MiM-in > DS google.com.
MiM-in > TXT google.com.
MiM-in > MX google.com.
13:25 wa /2026-04-docker-dns-test (main $=!V): make watch-parent-centric
DNS Watcher PC-in
PC-in > A google.com.
PC-in > TXT google.com.
PC-in > MX google.com.
client> dig @parent-centric _dmarc.google.com txt
[1] 0:docker*
"@489e58591910:/data" 13:29 03-Nov-26

```
13:25 wa /2026-04-docker-dns-test (main $=!V): make watch-mim

DNS Watcher MiM-in

MiM-in  >  NS .
MiM-in  >  NS com.
MiM-in  >  DNSKEY .
MiM-in  >  DNSKEY com.
MiM-in  >  A google.com.
MiM-in  >  DS google.com.
MiM-in  >  TXT google.com.
MiM-in  >  MX google.com.
MiM-in  >  TXT _dmarc.google.com.
```

```
13:25 wa /2026-04-docker-dns-test (main $=!V): make watch-parent-centric

DNS Watcher PC-in

PC-in   >  A google.com.
PC-in   >  TXT google.com.
PC-in   >  MX google.com.
PC-in   >  TXT _dmarc.google.com.
```

```
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: 8aef51e40def9150010000006aea5236745b9d013a667fcb (good)
;; QUESTION SECTION:
;_dmarc.google.com.             IN      TXT

;; ANSWER SECTION:
_dmarc.google.com.      300     IN      TXT     "v=DMARC1; p=none; sp=none; rua=mailto:elsewhere@smtp.evil.example"

;; Query time: 73 msec
;; SERVER: 10.200.0.3#53(parent-centric) (UDP)
;; WHEN: Tue Nov 03 21:29:26 UTC 2026
;; MSG SIZE  rcvd: 152

client>
```

```
[1] 0:docker*                                                    "@489e58591910:/data" 13:29 03-Nov-26
```

```
13:25 wa /2026-04-docker-dns-test (main $=!V): make watch-mim

DNS Watcher MiM-in

MiM-in  >  NS .
MiM-in  >  NS com.
MiM-in  >  DNSKEY .
MiM-in  >  DNSKEY com.
MiM-in  >  A google.com.
MiM-in  >  DS google.com.
MiM-in  >  TXT google.com.
MiM-in  >  MX google.com.
MiM-in  >  TXT _dmarc.google.com.
```

```
13:25 wa /2026-04-docker-dns-test (main $=!V): make watch-parent-centric

DNS Watcher PC-in

PC-in   >  A google.com.
PC-in   >  TXT google.com.
PC-in   >  MX google.com.
PC-in   >  TXT _dmarc.google.com.
```

```
client> dig @parent-centric www.google.com
```

```
[1] 0:docker*                                "@489e58591910:/data" 13:30 03-Nov-26
```

```
13:25 wa /2026-04-docker-dns-test (main $=!V): make watch-mim

DNS Watcher MiM-in

MiM-in  >  NS .
MiM-in  >  NS com.
MiM-in  >  DNSKEY .
MiM-in  >  DNSKEY com.
MiM-in  >  A google.com.
MiM-in  >  DS google.com.
MiM-in  >  TXT google.com.
MiM-in  >  MX google.com.
MiM-in  >  TXT _dmarc.google.com.
MiM-in  >  A www.google.com.
```

```
13:25 wa /2026-04-docker-dns-test (main $=!V): make watch-parent-centric

DNS Watcher PC-in

PC-in   >  A google.com.
PC-in   >  TXT google.com.
PC-in   >  MX google.com.
PC-in   >  TXT _dmarc.google.com.
PC-in   >  A www.google.com.
```

```
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: 34c34a4d3e073914010000006aea5282c19422e75a0d15e3 (good)
;; QUESTION SECTION:
;www.google.com.                        IN      A

;; ANSWER SECTION:
www.google.com.         300     IN      A       6.6.6.6

;; Query time: 73 msec
;; SERVER: 10.200.0.3#53(parent-centric) (UDP)
;; WHEN: Tue Nov 03 21:30:42 UTC 2026
;; MSG SIZE  rcvd: 87

client>
```

```
[1] 0:docker*                                   "@489e58591910:/data" 13:30 03-Nov-26
```

So What?

Ramifications of MiM Injections

TL;DR: significantly increased attack surface

1. Visibility of DNS requests normally out of view

2. Modification of DNS responses normally out of view

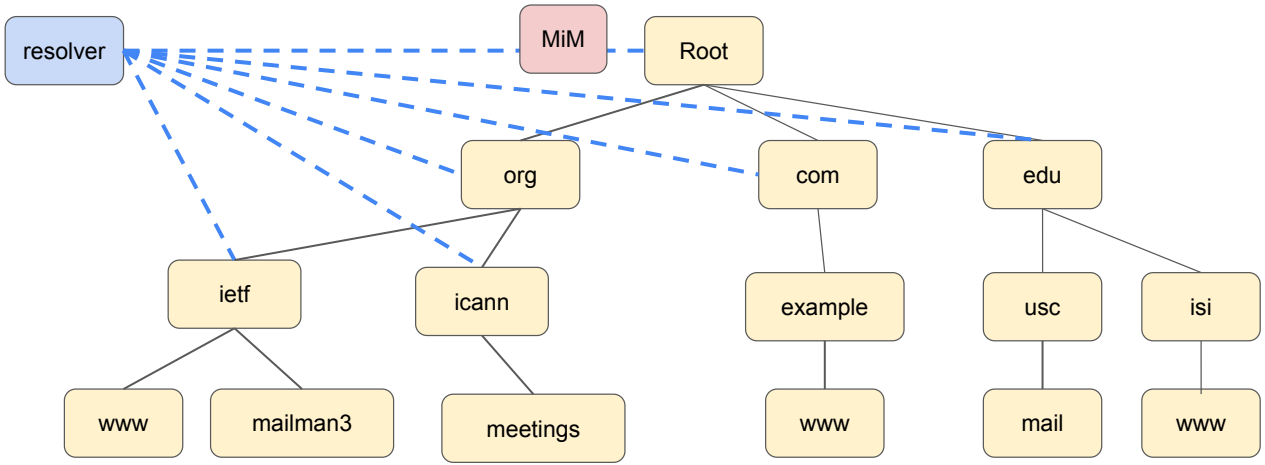
Let’s look at someone else’s DNS requests





MiM in only one path can provide **100% DNS visibility**

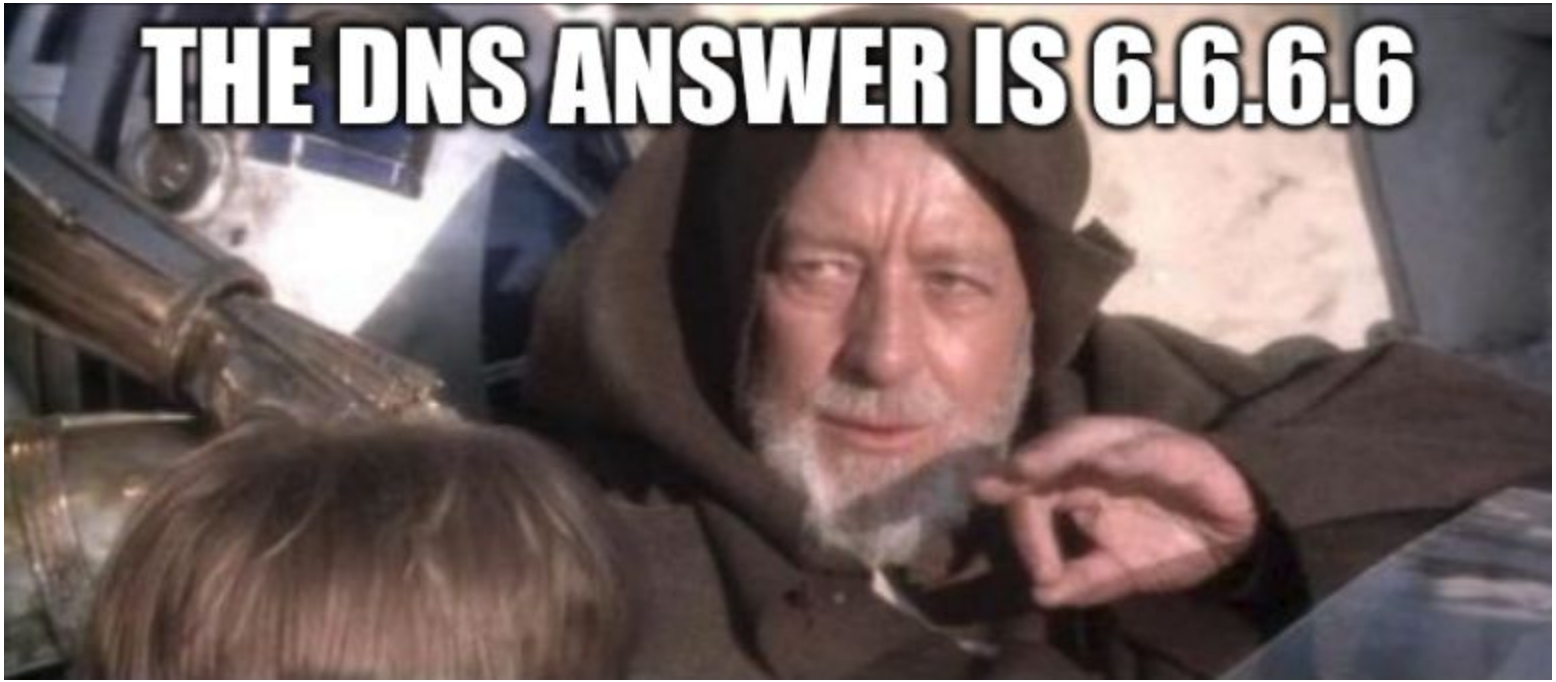
Even though **ALL** these zones are **DNSSEC signed**





Parent-centric ramifications of not checking DNSSEC

- Additional MiM vantage points can be inserted into the DNS tree
  - Starting with any vantage point higher in the tree
- Difficult to detect in the client
- A MiM can:
  - Modify delegation information
    - A / AAAA glue *(this is hard to detect)*
    - NS names *(this is easier to detect)*
  - View all DNS transactions beyond their normal vantage point
  - Modify unsigned contents lower in the tree they normally can't see


THE DNS ANSWER IS 6.6.6.6

Attacking normally off-path, non-DNSSEC protected zones

The new expanded vantage points covering non-DNSSEC protected zones:

- Alter MX records
  - Re-routing mail through a attacker controlled relay
- Alter DKIM/DMARC
  - Spoofing mail to the attack target from other DNS services
- Alter DNS HTTPS records
  - Downgrading http version
  - ECH manipulation
  - Adding or changing ipv4/v6 glue
- …

FAQ

Does client-centric resolution solve this?

Yes: because the child verifies delegation information *(caveat: some give up)*

Can the end-user / client detect this?

Not easily: the delegation cache is different than the cache for the client

Can DNSSEC prevent this?

No: because parent-side NS/glue are never checked

But: Data to clients is validated: DS -> DNSKEY -> … -> DNSKEY -> RR

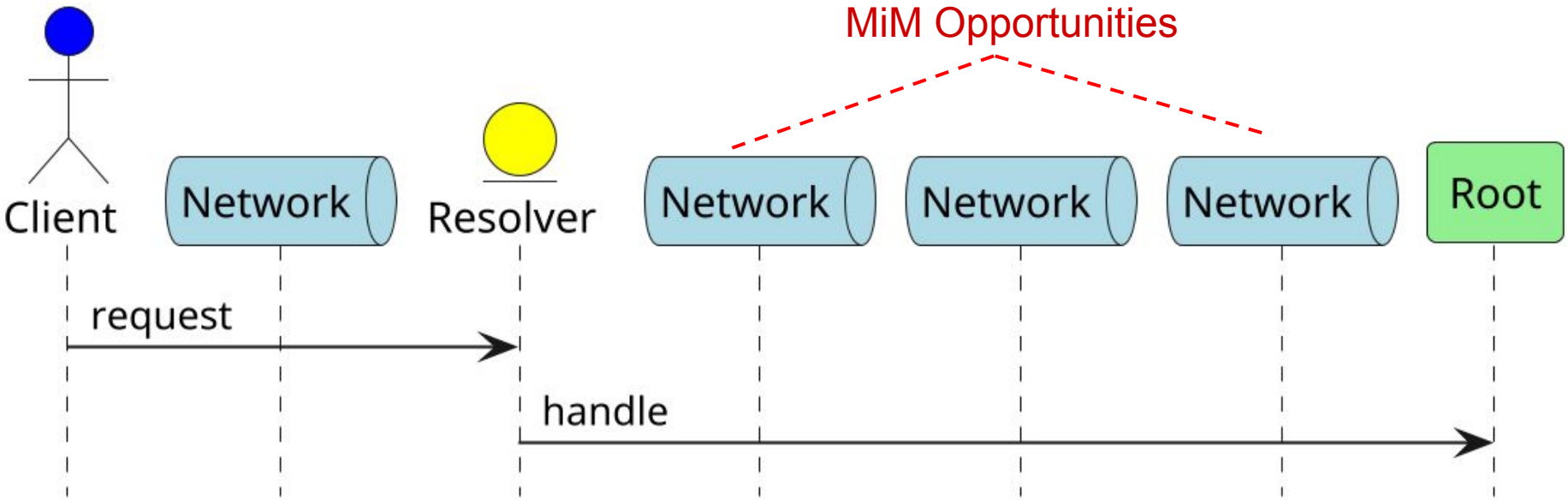
What about DELEG?

DELEG delegations are secure: as they'll be signed by the parent

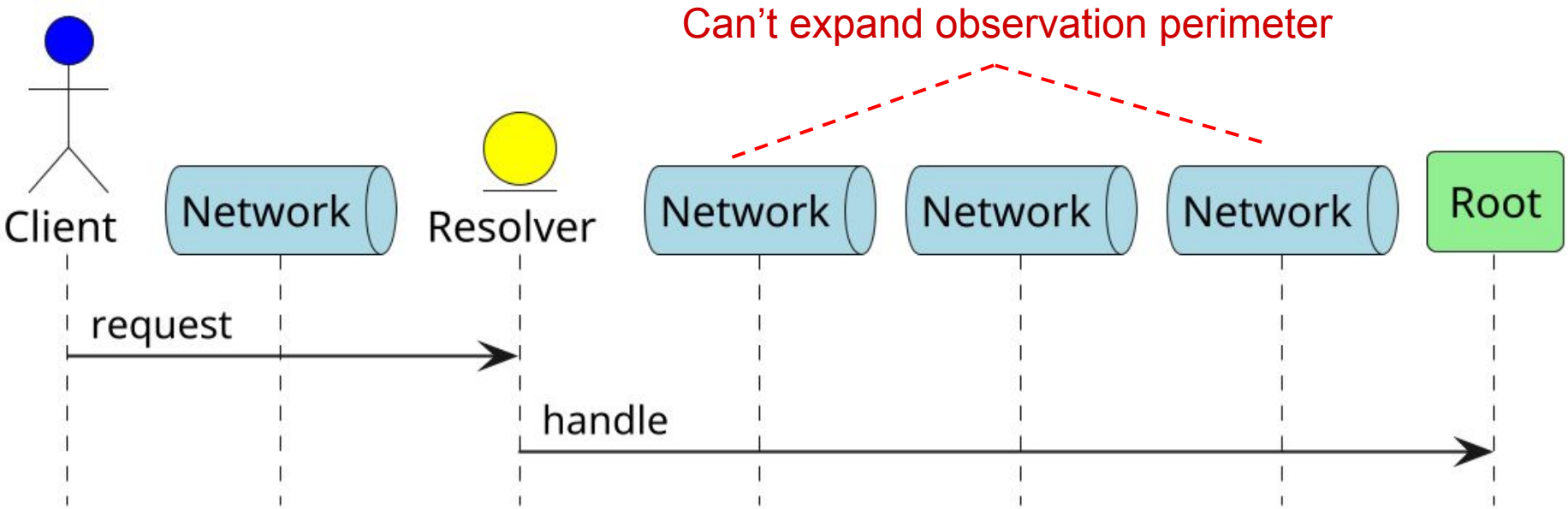
Mitigations

- Just say no to parent-centric, validating resolvers
  - Use DNSSEC validating, child-centric resolvers
  - See draft-ietf-dnsop-ns-revalidation
- Deploy parent-centric resolvers close to authoritative server infrastructure
  - aka peer widely: minimize MiM opportunities
- Deploy DoX between your clients and your resolver
- Look forward to DELEG records, which are parent-signed

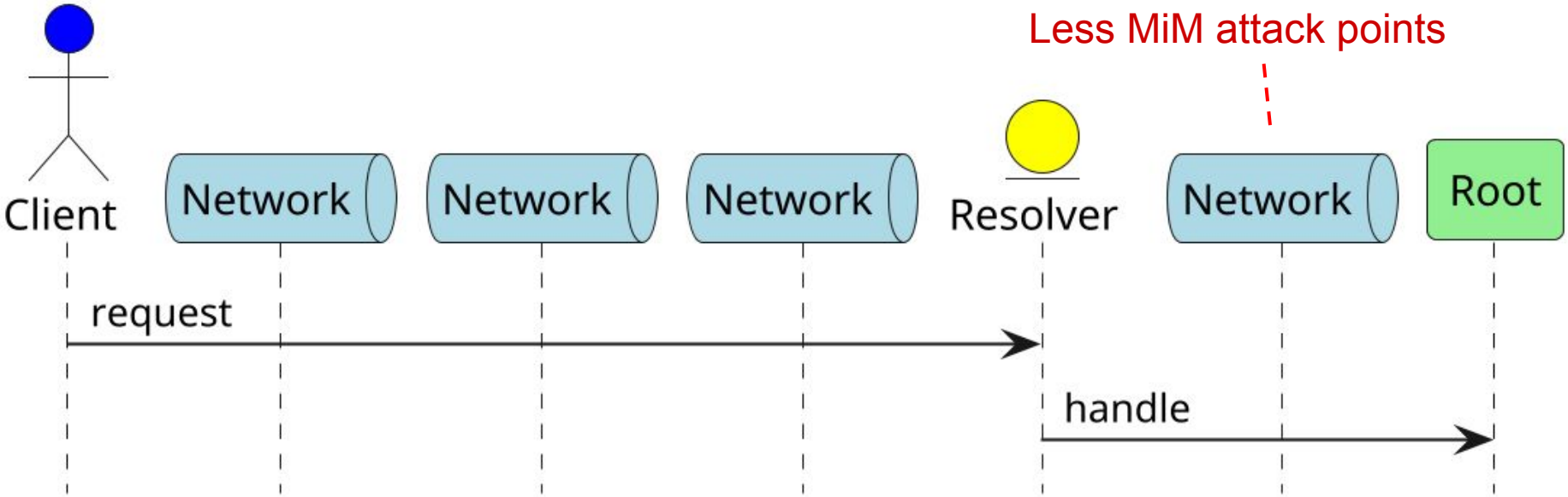
Resolvers are typically placed near clients

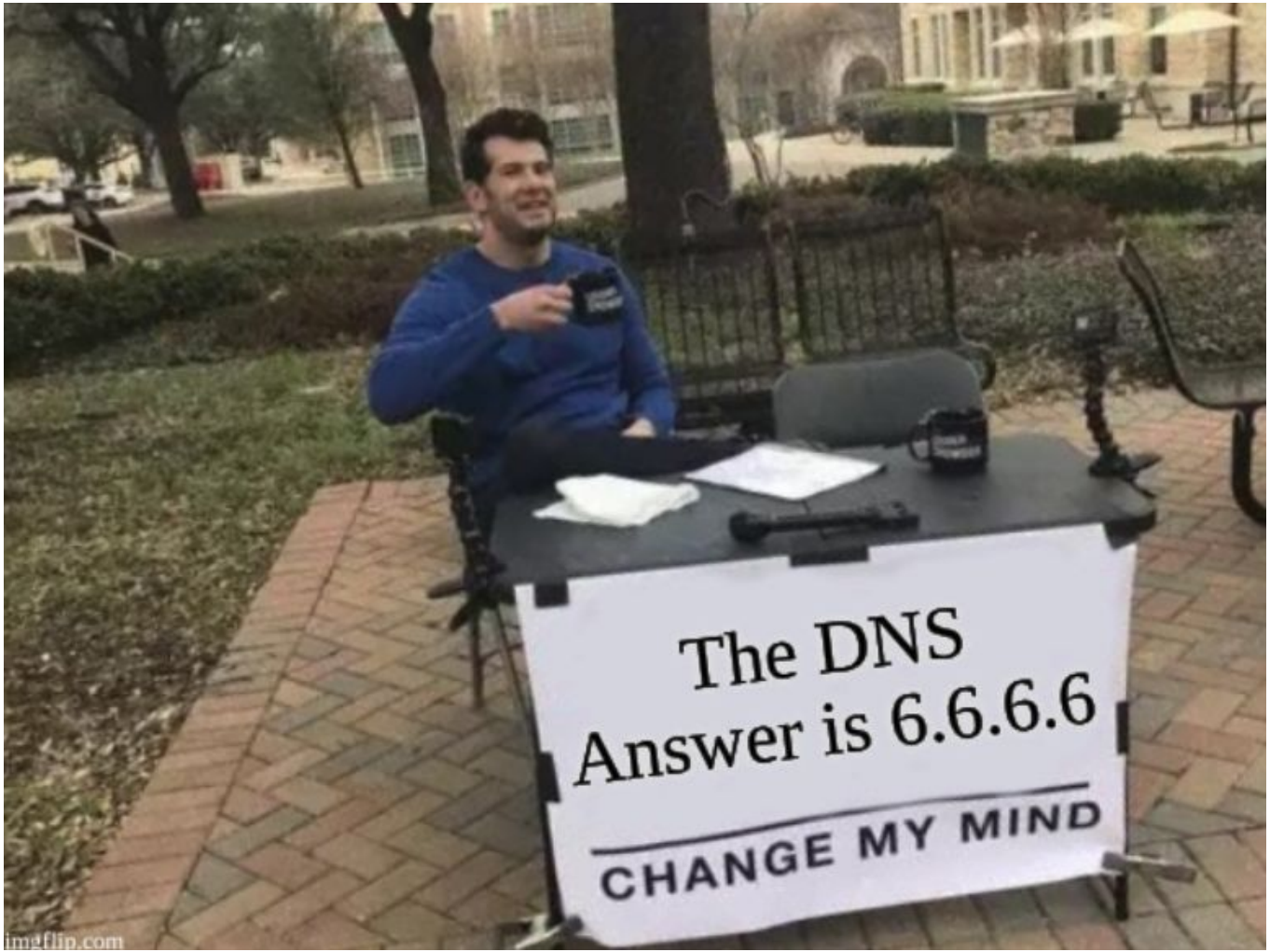


Option 1: deploy client-centric, validating resolvers



Option 2: place resolvers closer to authoritative servers

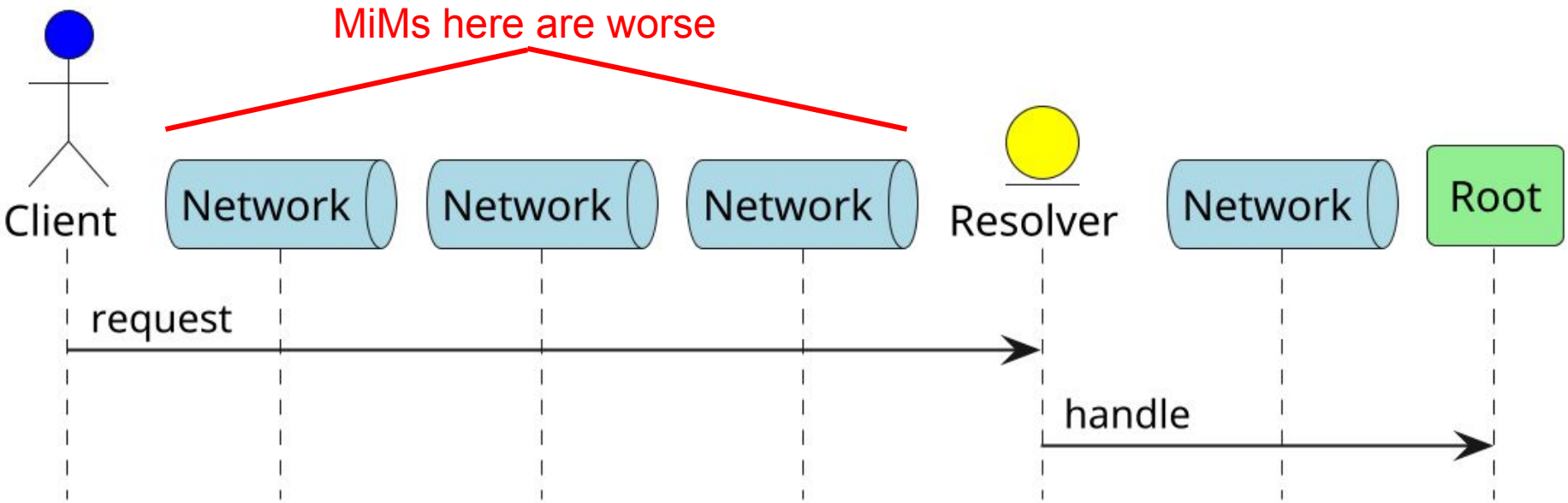




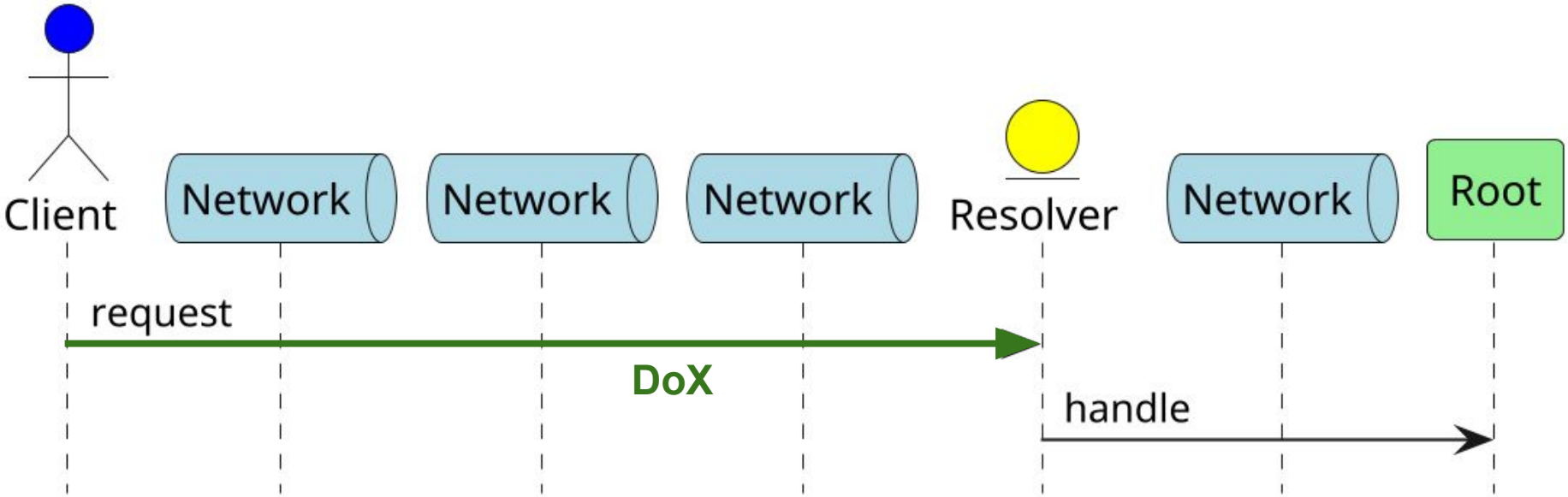
Thanks for the new MiM spot

You've only moved the problem

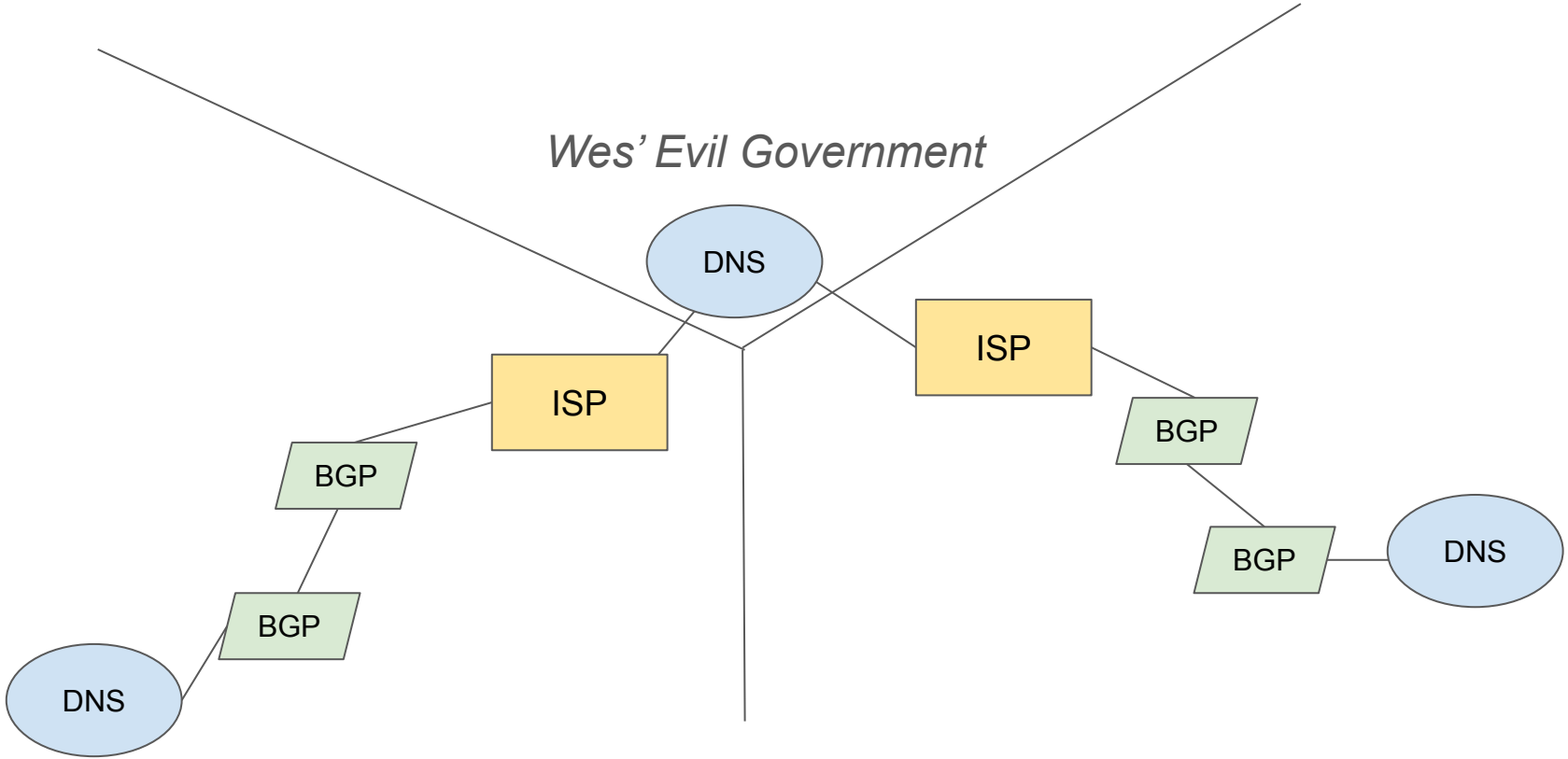
You've just moved the problem



Solution: deploy DoT here



Just one more thing



Takeaways

- Parent-centric, DNSSEC validating resolvers:
  - Can be subject to expanded **view injection attacks** for all zones
  - Can be subject to **expanded modification attacks** for unsigned zones
  - Other work has shown them to be faster and more robust, however

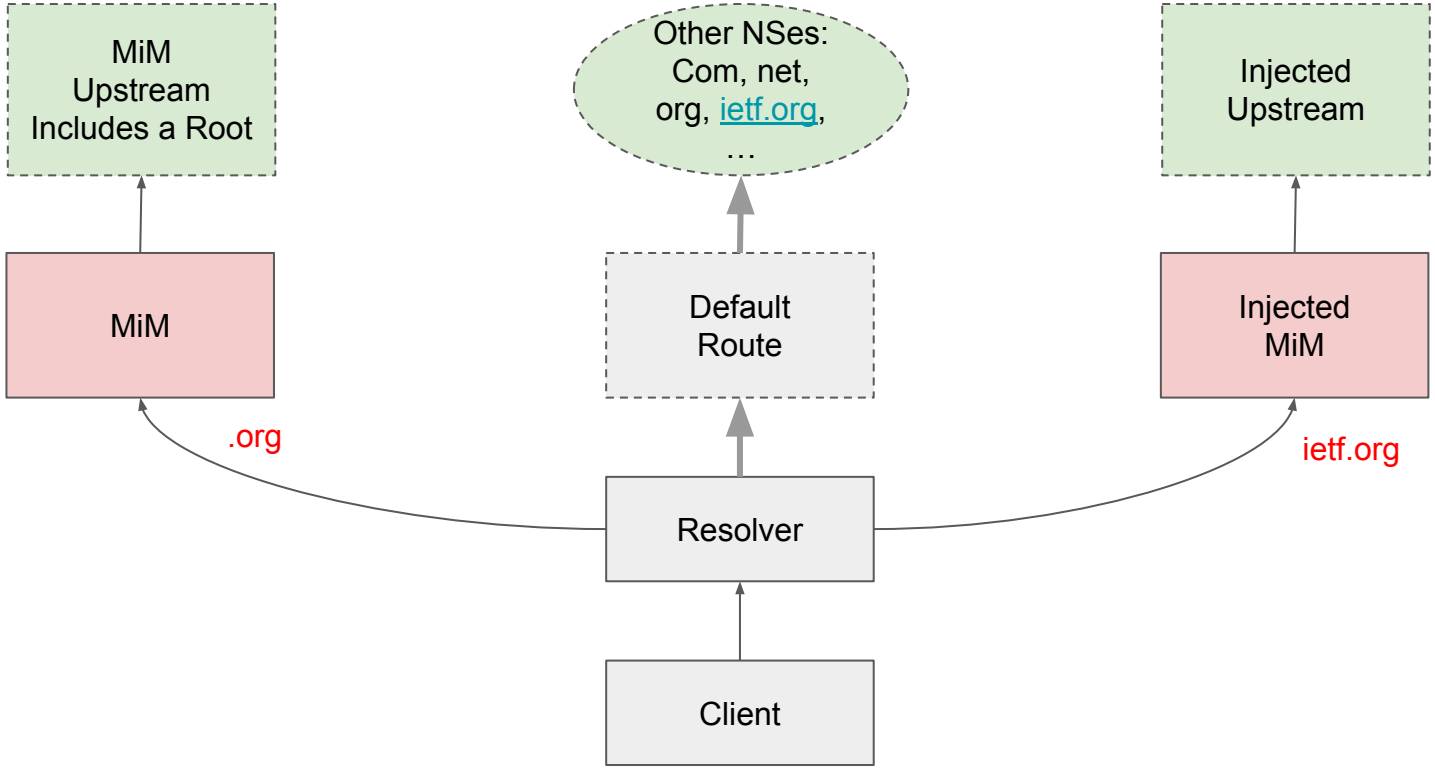
- **Mitigations:** *what are you going to prioritize? Security? Robustness?*
  - **Choose** resolver software **carefully**
  - Only deploy parent-centric resolvers with **wide-peering**

- DELEG can’t come fast enough

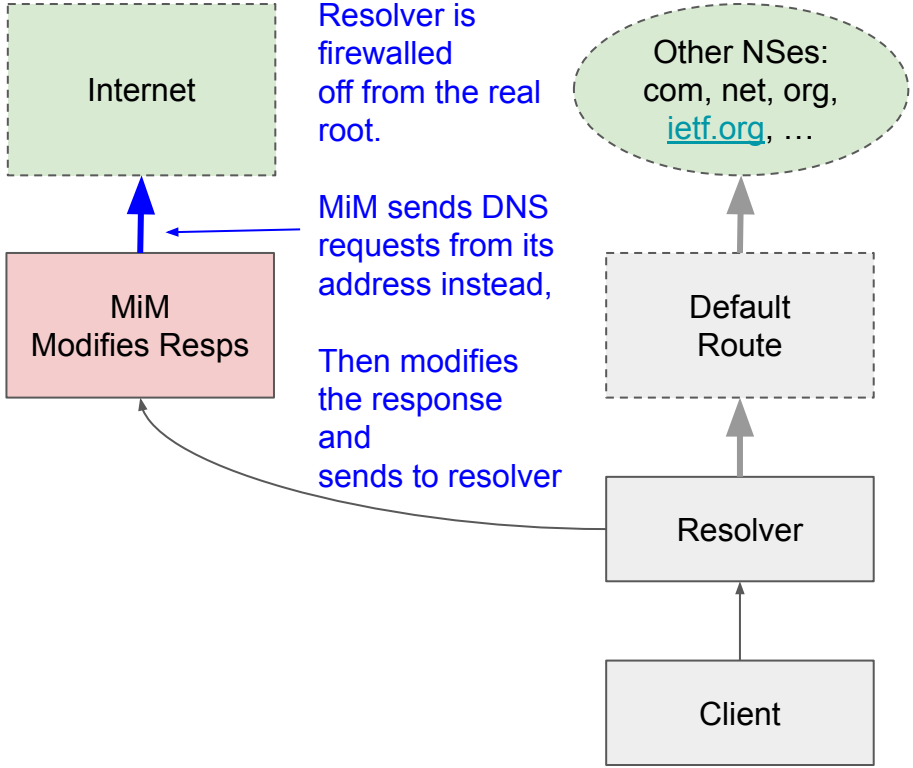
Questions? Contact: hardaker@{google.com, isi.edu, hardakers.net}

Backup Extra Complex Slides

Note: other MiMs can be injected too



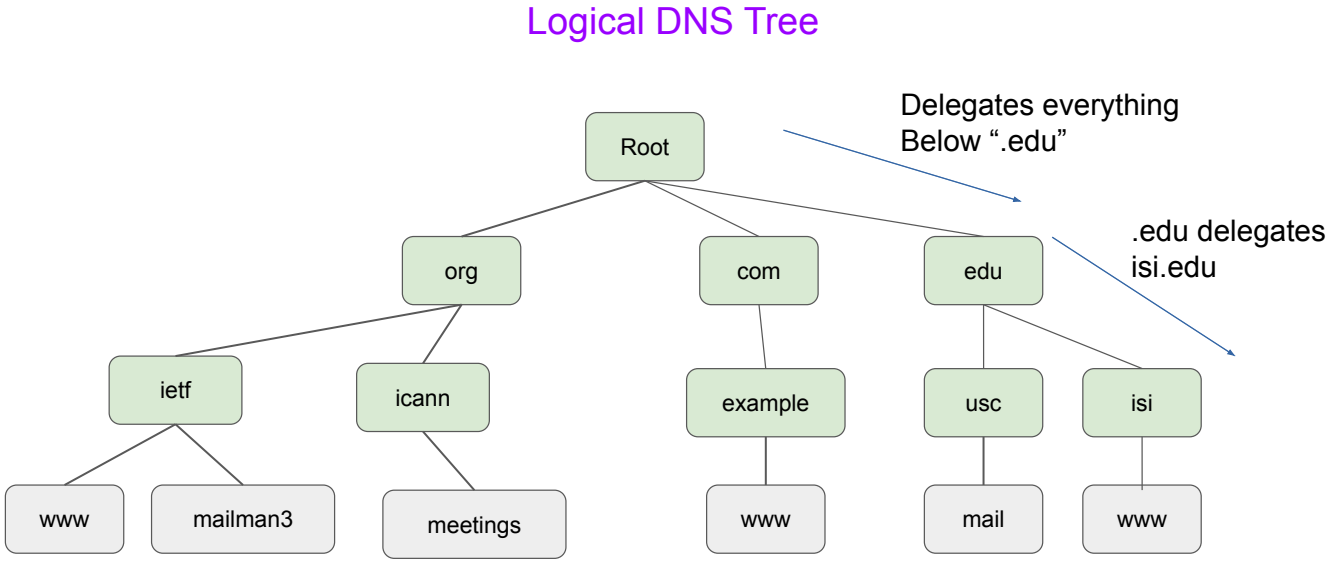
Implementation Notes:



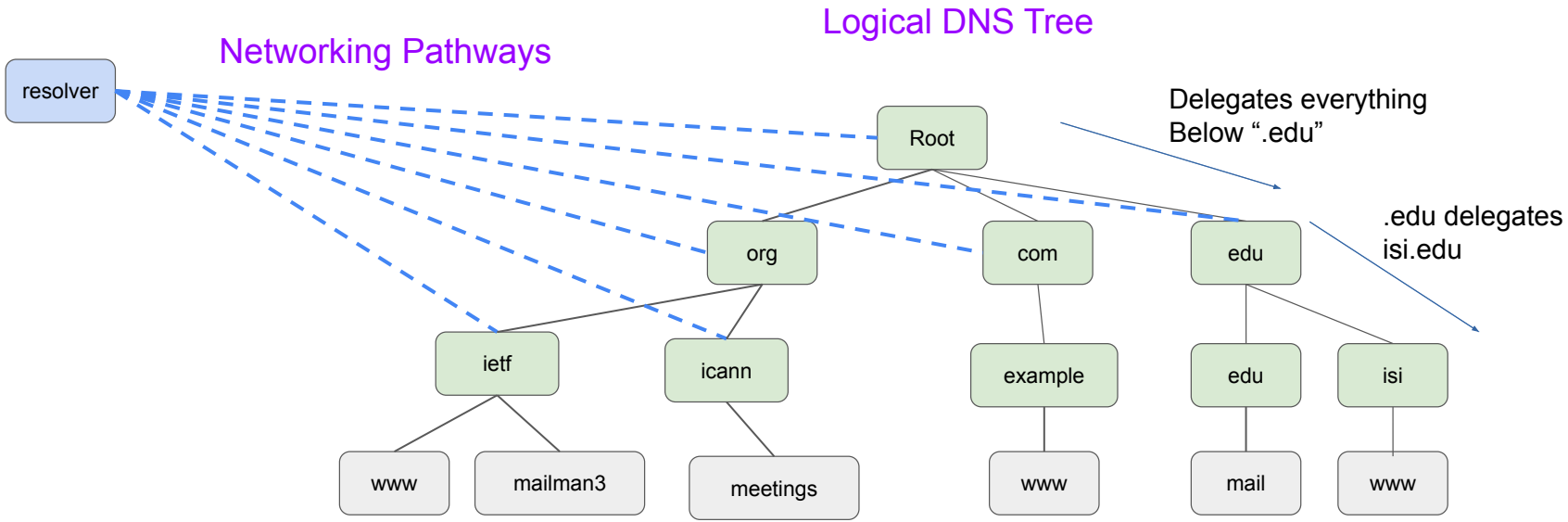
Parent Cache

Child Cache

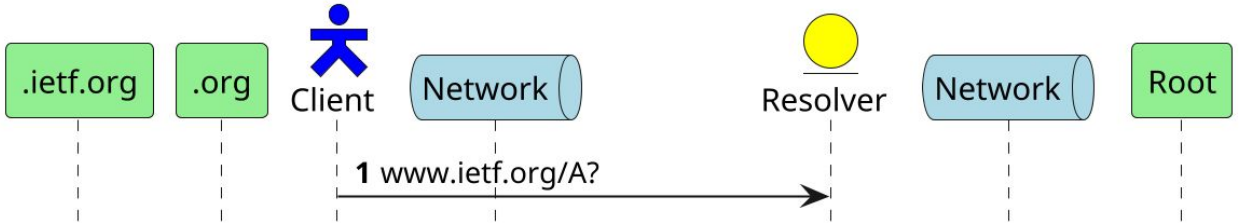
DNS Resolvers Communicate With Everything



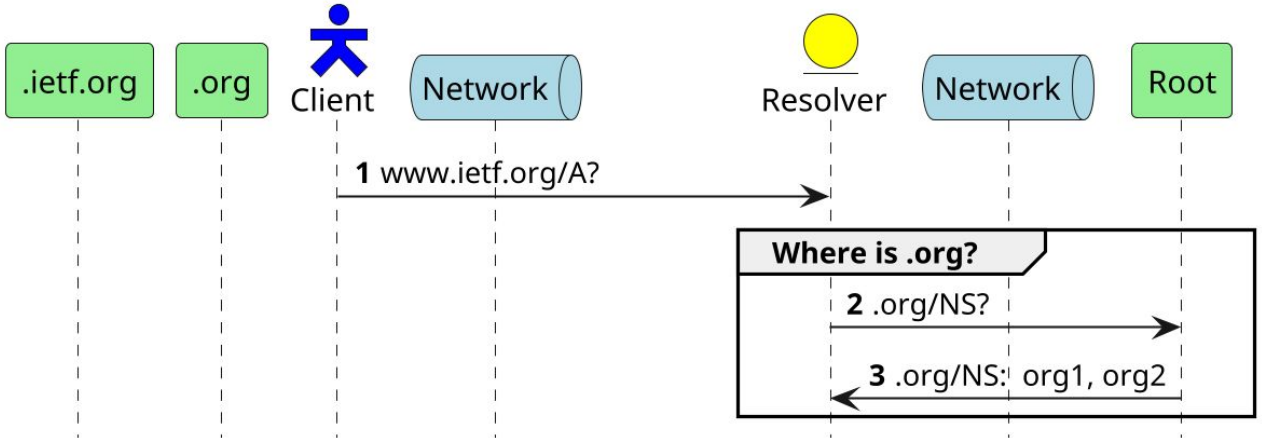
DNS Resolvers Communicate With Everything



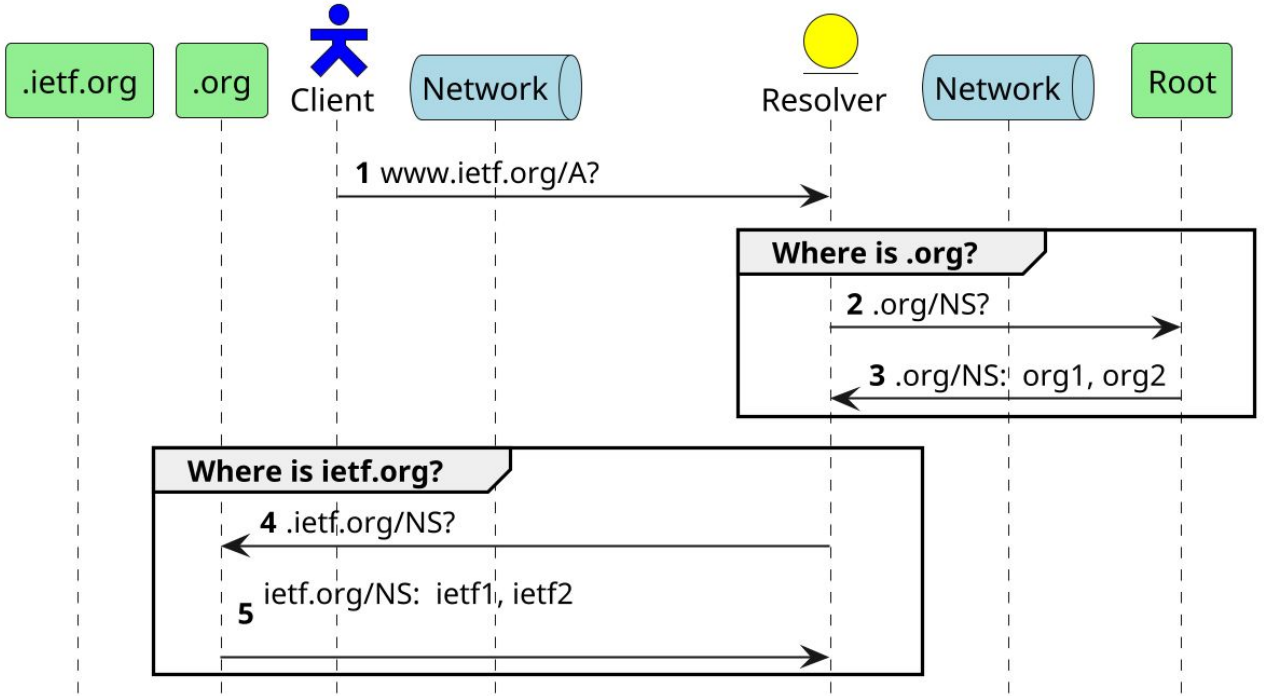
DNS Resolution Steps



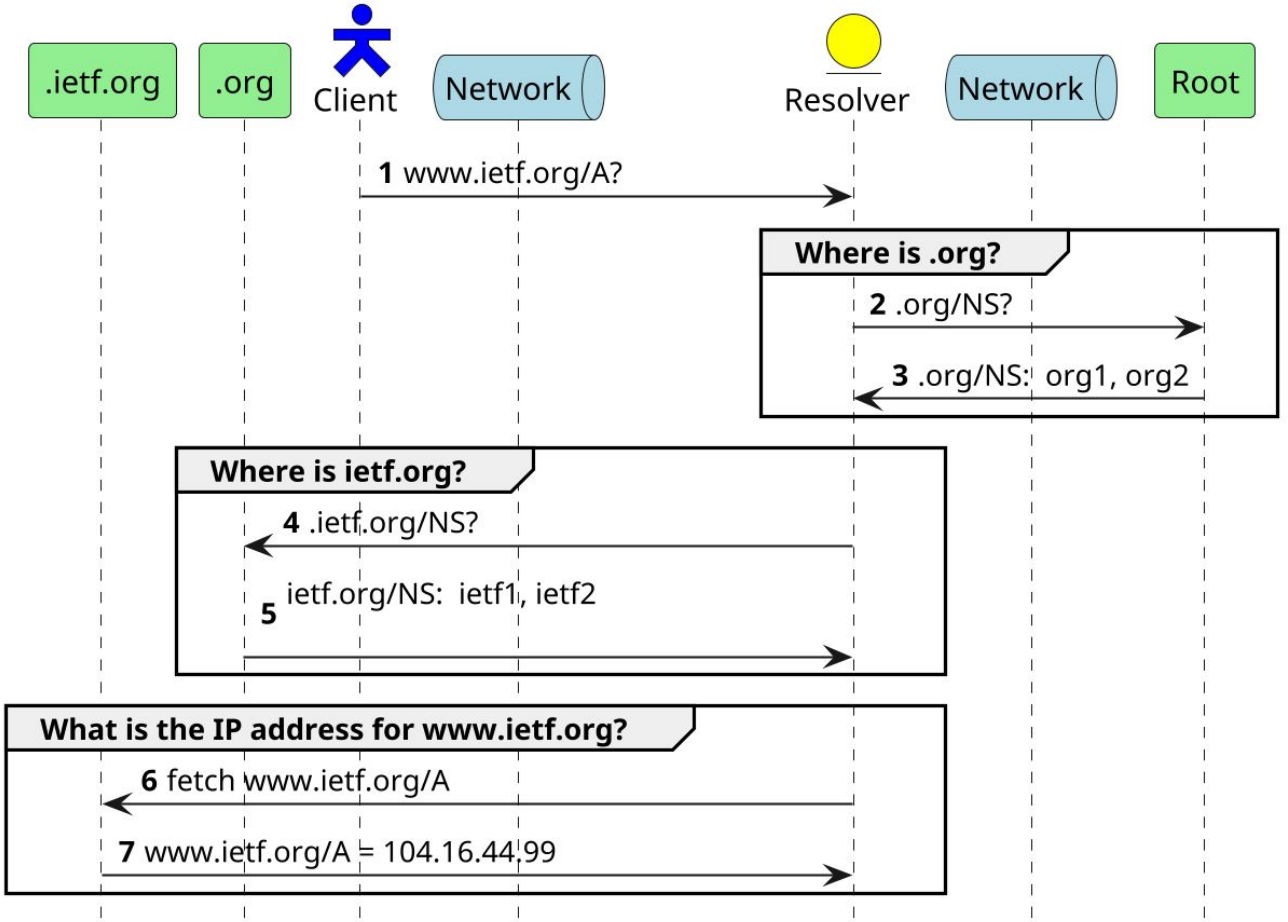
DNS Resolution Steps



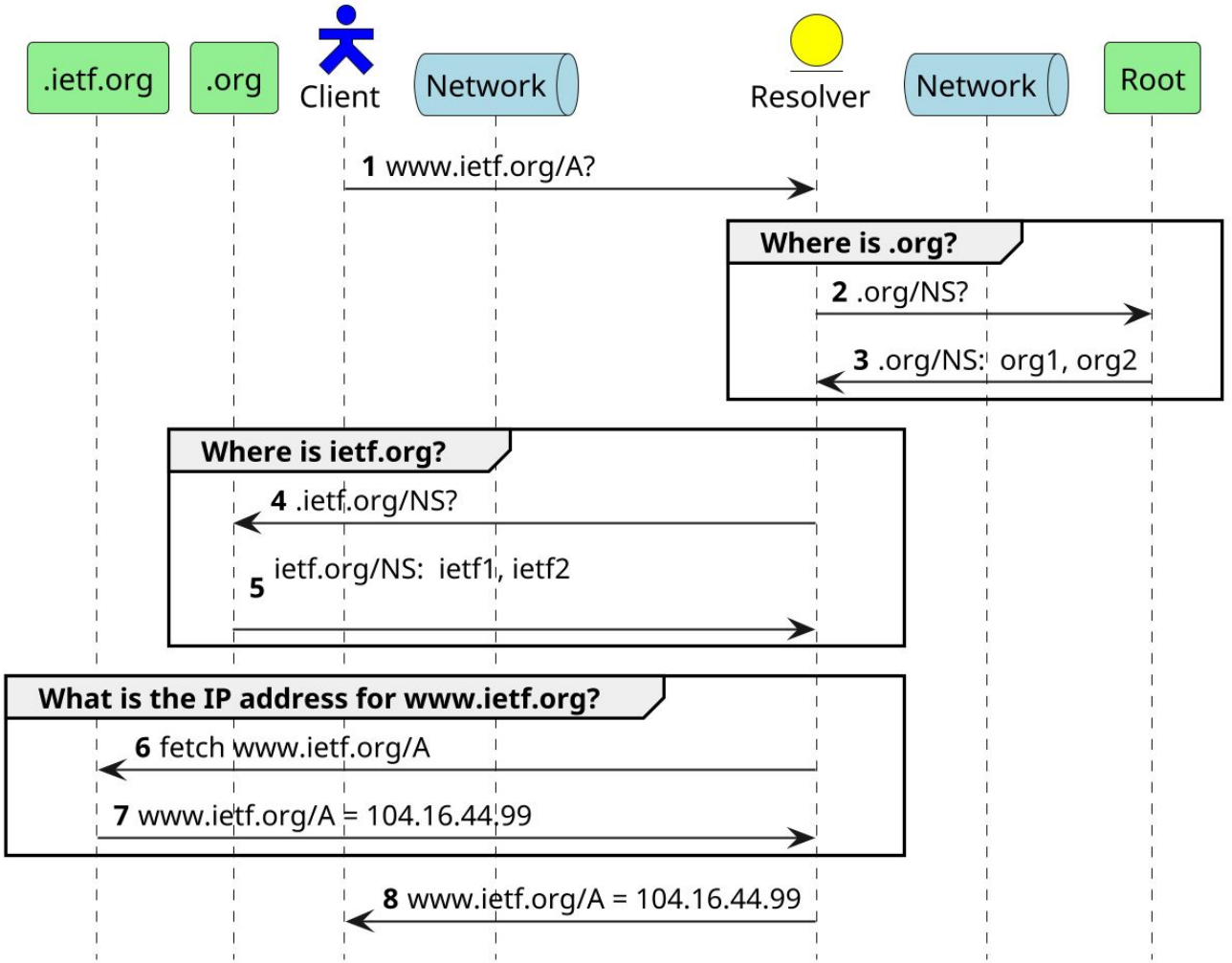
DNS Resolution Steps



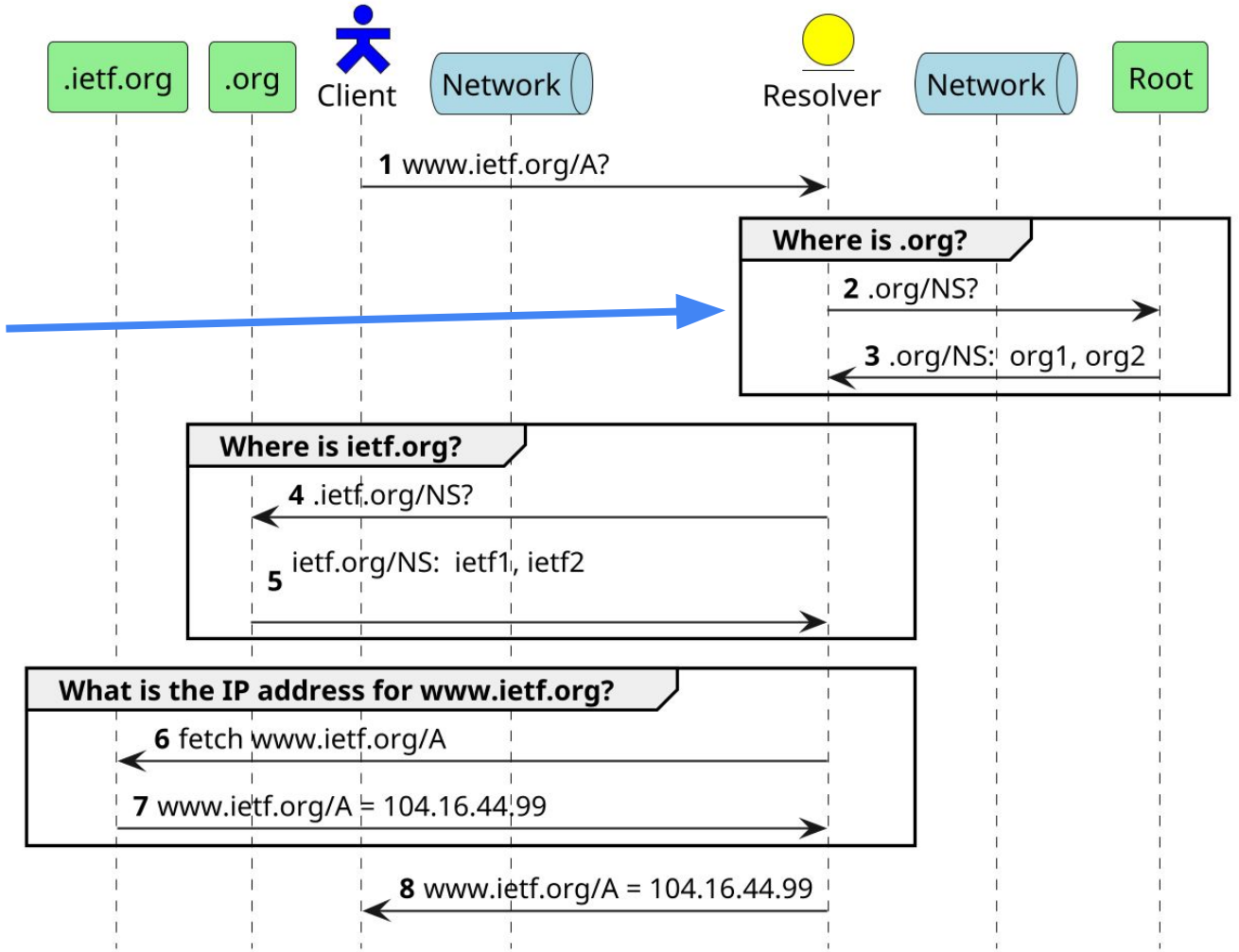
DNS Resolution Steps



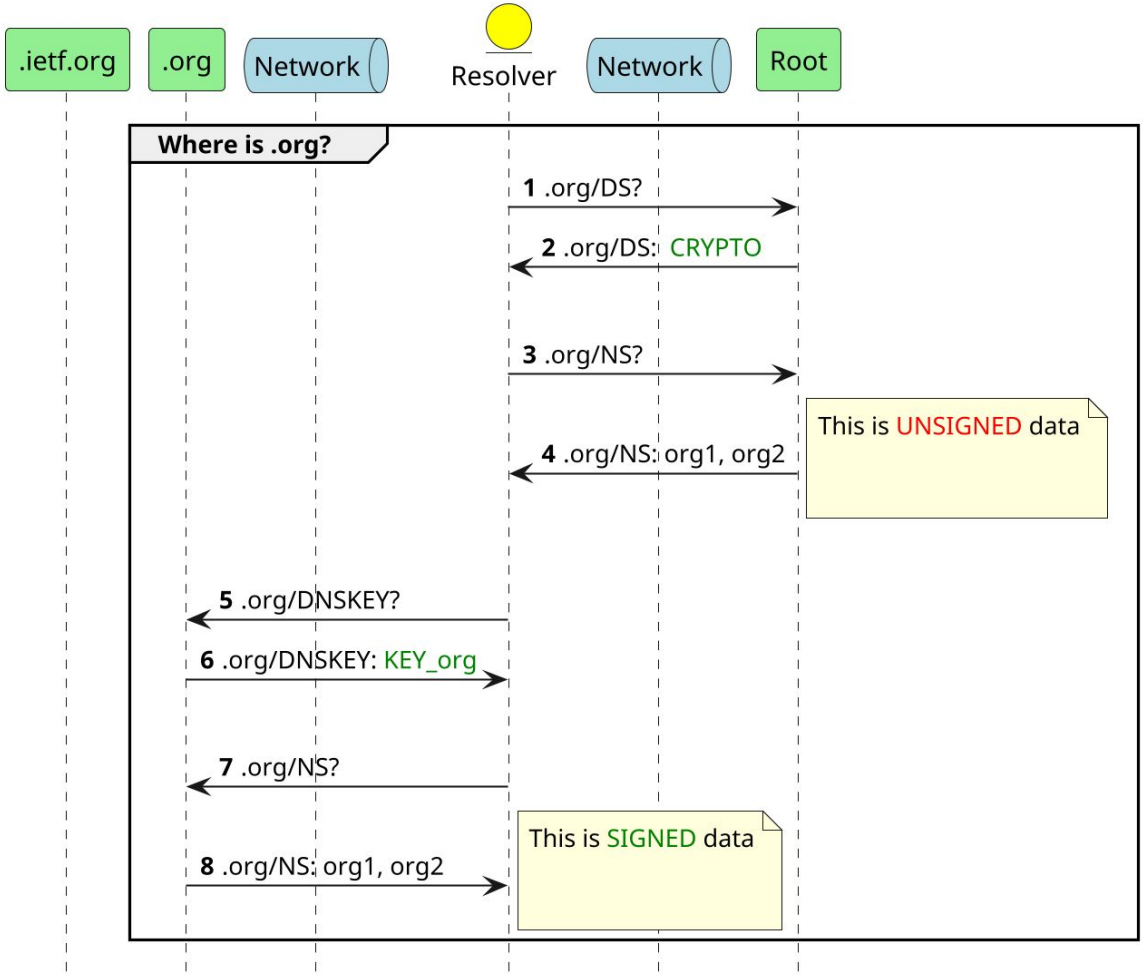
DNS Resolution Steps



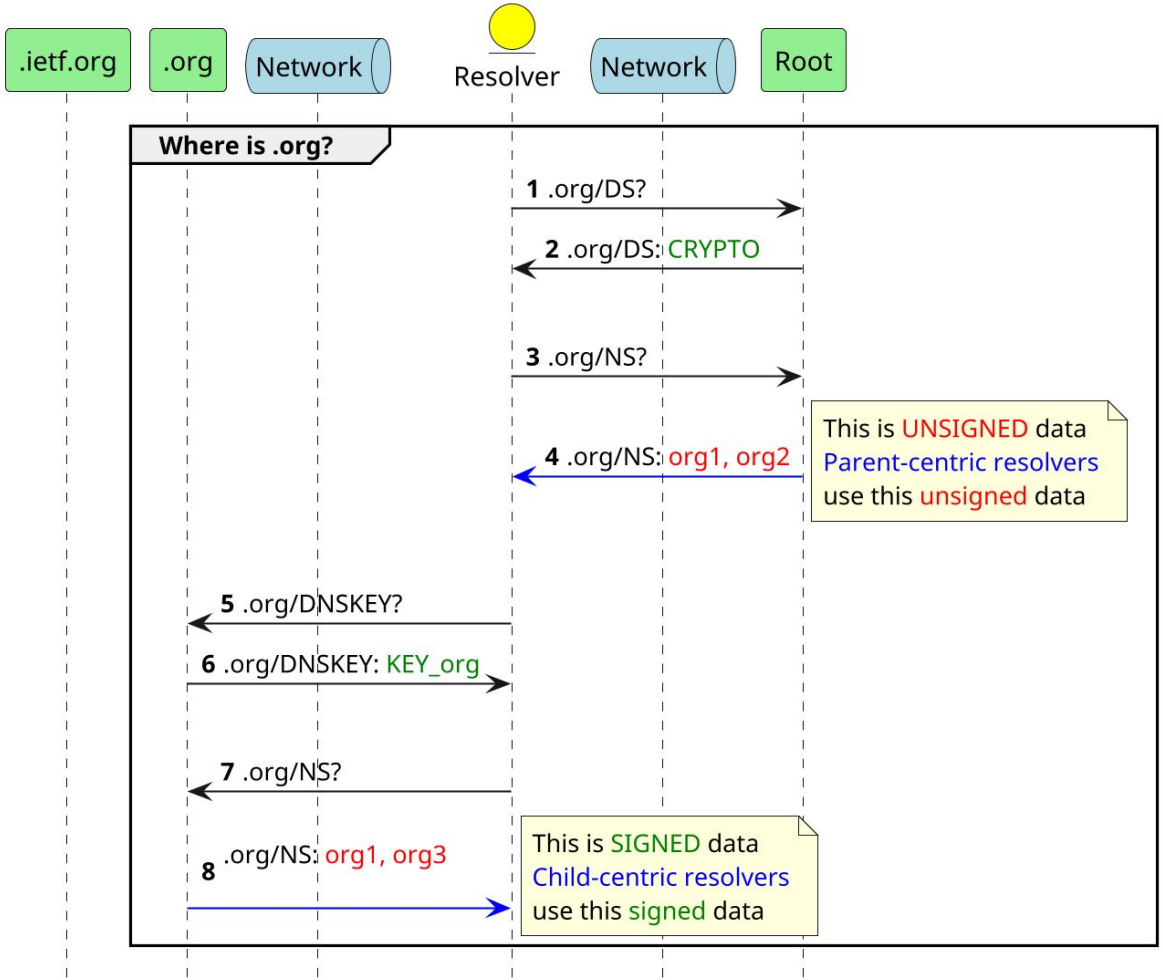
Let's dive into just this transaction with DNSSEC validation

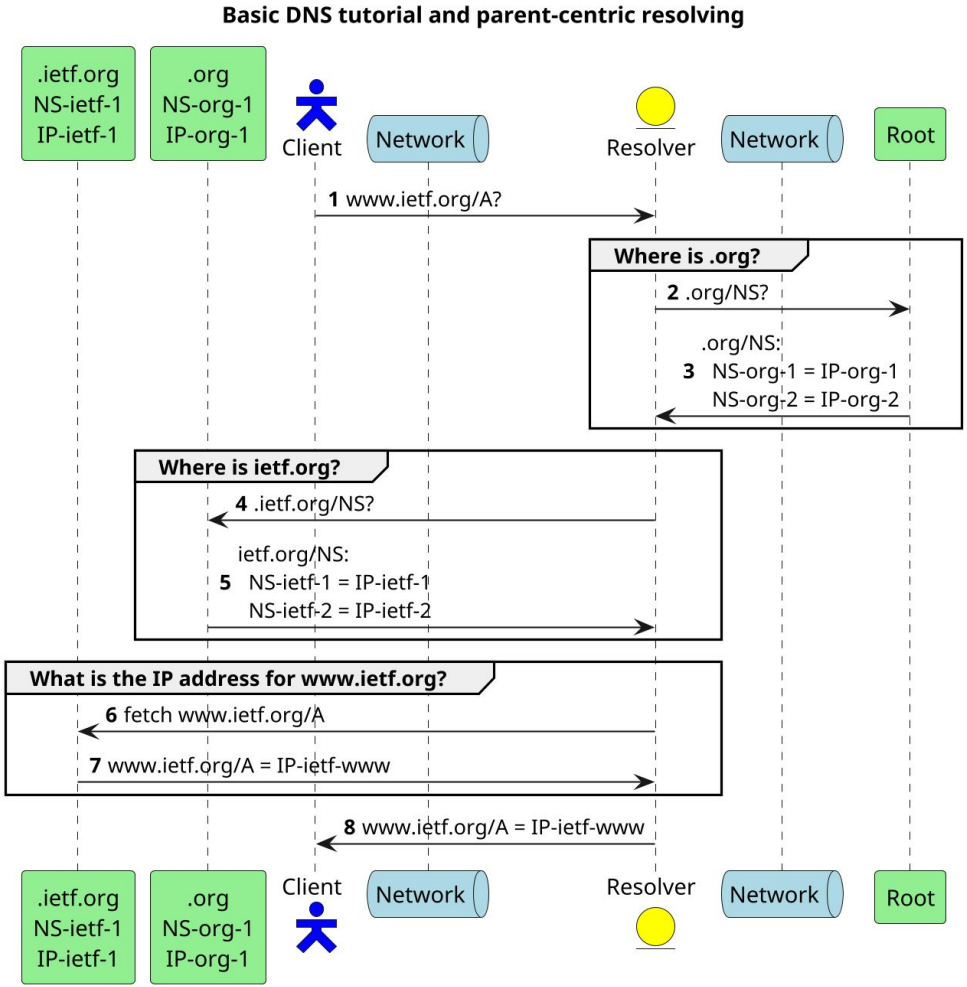


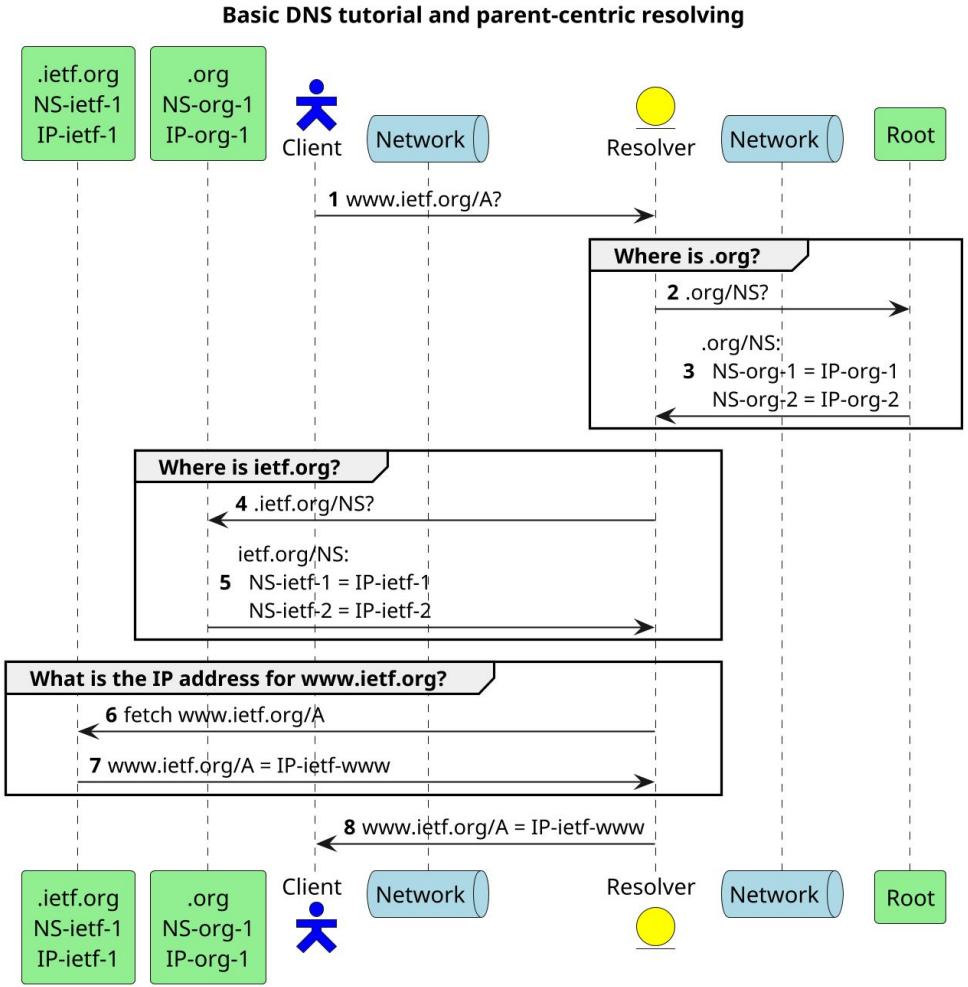
DNSSEC Validation of .org data



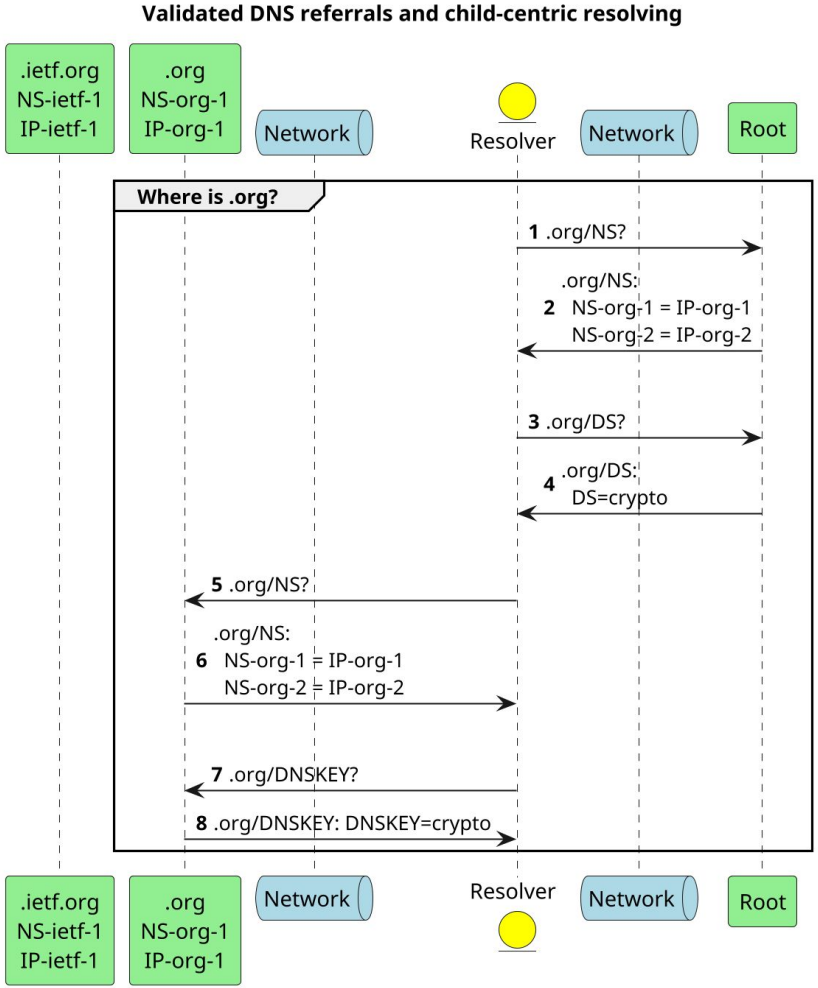
What Happens When Differences Arise?



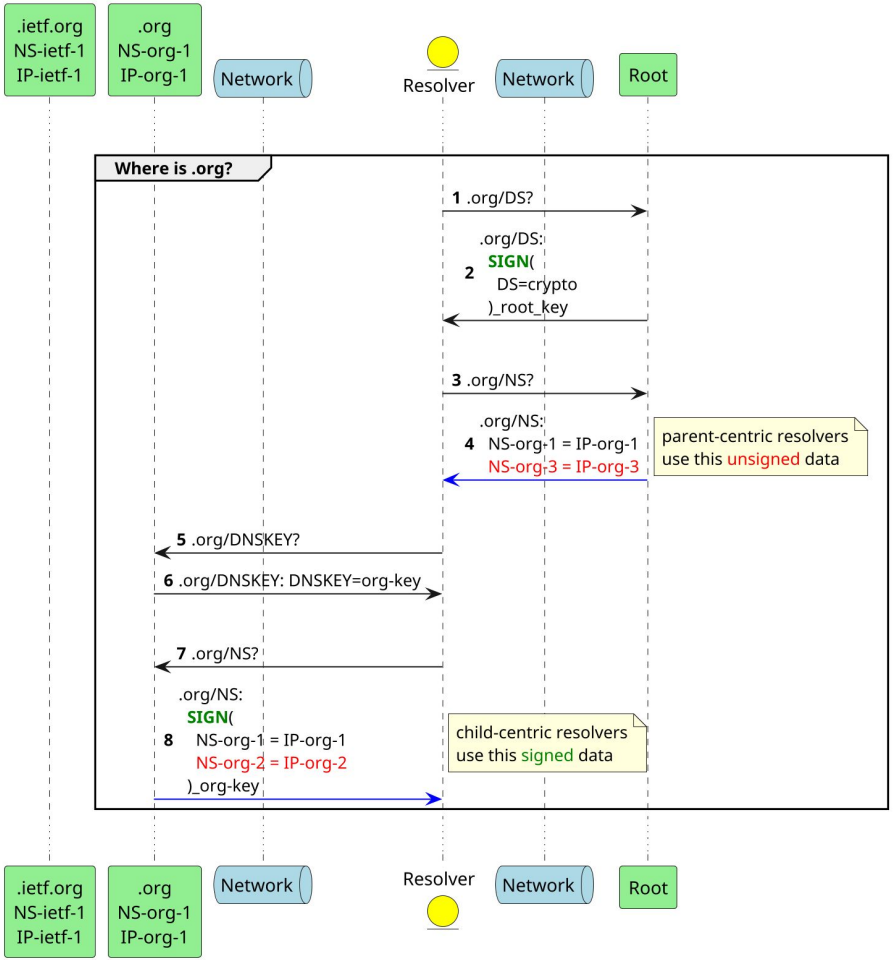

Basic DNS tutorial and parent-centric resolving
.ietf.org
NS-ietf-1
IP-ietf-1
.org
NS-org-1
IP-org-1
Client
Network
Resolver
Network
Root
1 www.ietf.org/A?
Where is .org?
2 .org/NS?
.org/NS:
3 NS-org-1 = IP-org-1
NS-org-2 = IP-org-2
Where is ietf.org?
4 .ietf.org/NS?
ietf.org/NS:
5 NS-ietf-1 = IP-ietf-1
NS-ietf-2 = IP-ietf-2
What is the IP address for www.ietf.org?
6 fetch www.ietf.org/A
7 www.ietf.org/A = IP-ietf-www
8 www.ietf.org/A = IP-ietf-www
.ietf.org
NS-ietf-1
IP-ietf-1
.org
NS-org-1
IP-org-1
Client
Network
Resolver
Network
Root

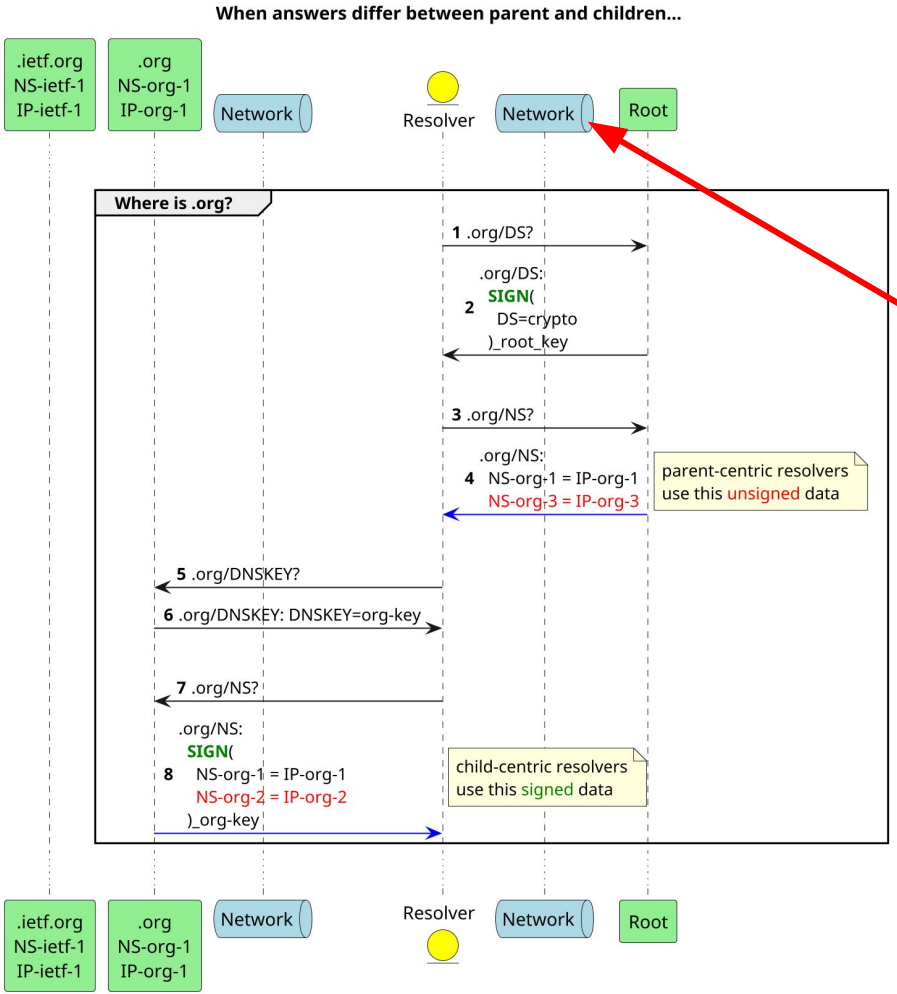


Let's dive into just this transaction with DNSSEC validation

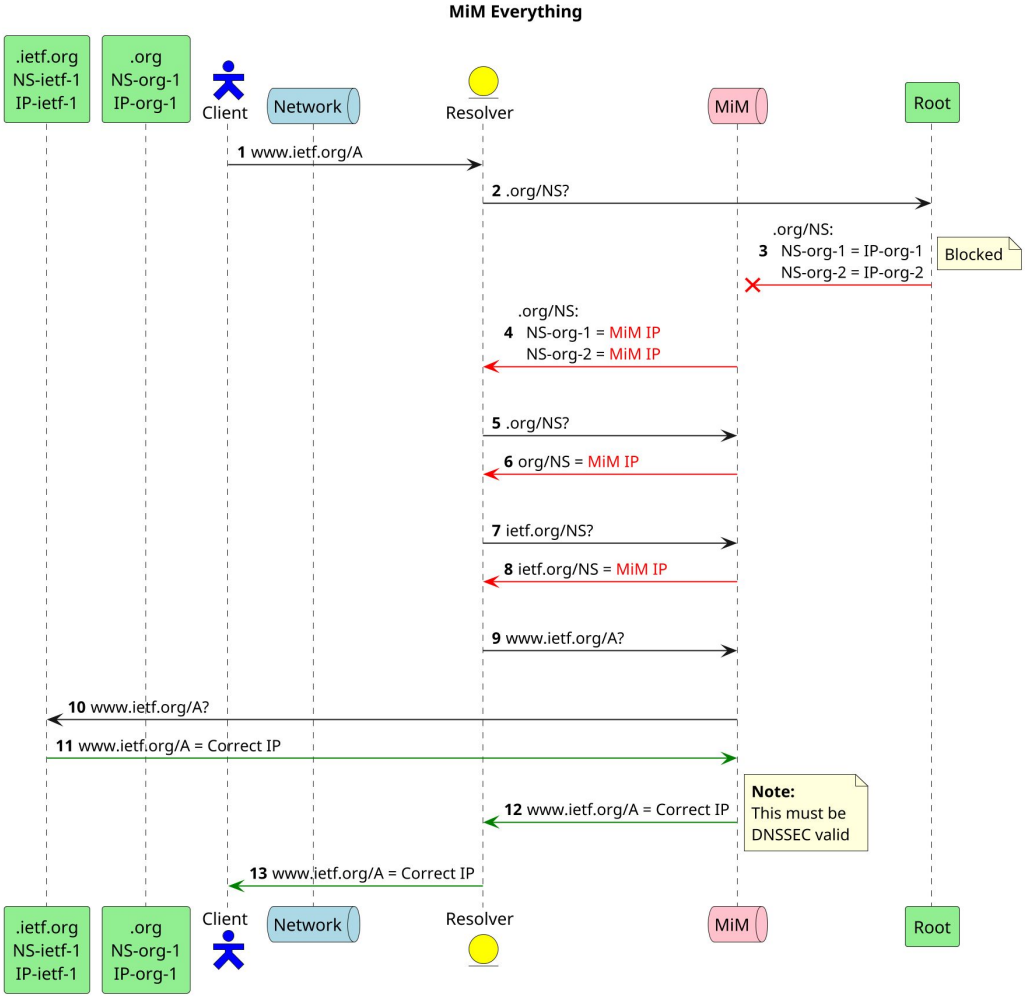

Validated DNS referrals and child-centric resolving
.ietf.org
NS-ietf-1
IP-ietf-1
.org
NS-org-1
IP-org-1
Network
Resolver
Network
Root
Where is .org?
1 .org/NS?
.org/NS:
2 NS-org-1 = IP-org-1
NS-org-2 = IP-org-2
3 .org/DS?
.org/DS:
4 DS=crypto
5 .org/NS?
.org/NS:
6 NS-org-1 = IP-org-1
NS-org-2 = IP-org-2
7 .org/DNSKEY?
8 .org/DNSKEY: DNSKEY=crypto
.ietf.org
NS-ietf-1
IP-ietf-1
.org
NS-org-1
IP-org-1
Network
Resolver
Network
Root

When answers differ between parent and children...

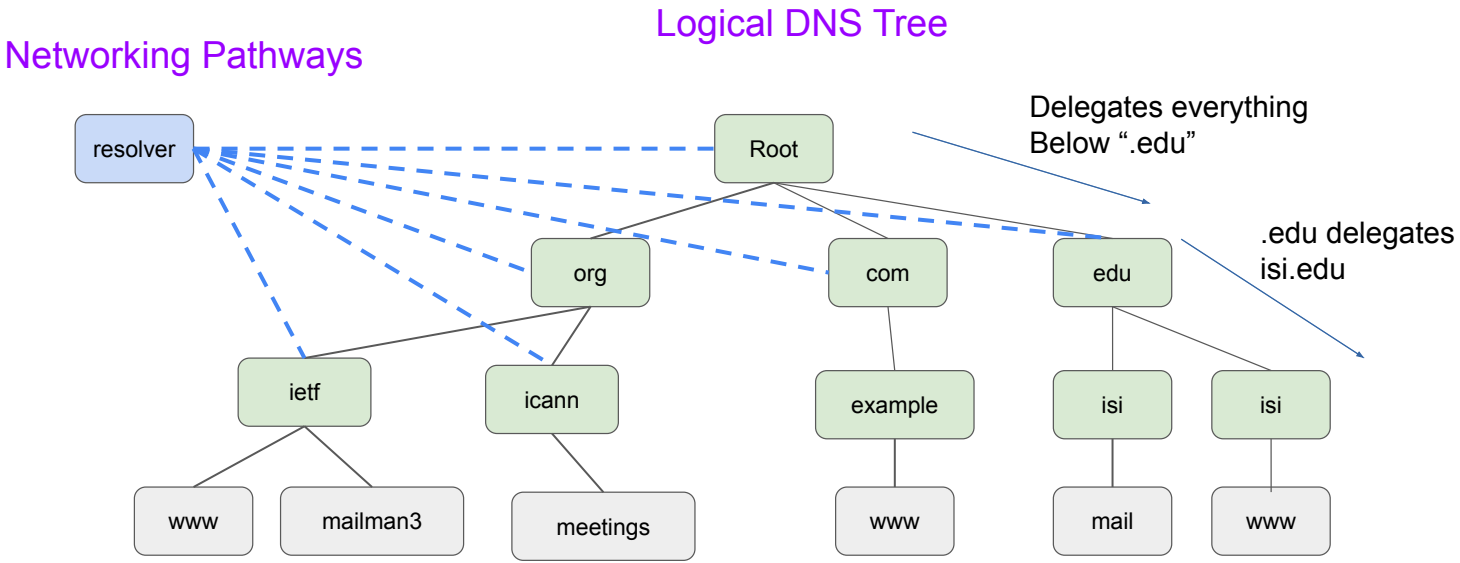




What if there was a MiM here?


MiM Everything
.ietf.org
NS-ietf-1
IP-ietf-1
.org
NS-org-1
IP-org-1
Client
Network
Resolver
MiM
Root
1 www.ietf.org/A
2 .org/NS?
.org/NS:
3 NS-org-1 = IP-org-1
NS-org-2 = IP-org-2
Blocked
.org/NS:
4 NS-org-1 = MiM IP
NS-org-2 = MiM IP
5 .org/NS?
6 org/NS = MiM IP
7 ietf.org/NS?
8 ietf.org/NS = MiM IP
9 www.ietf.org/A?
10 www.ietf.org/A?
11 www.ietf.org/A = Correct IP
Note:
This must be
DNSSEC valid
12 www.ietf.org/A = Correct IP
13 www.ietf.org/A = Correct IP

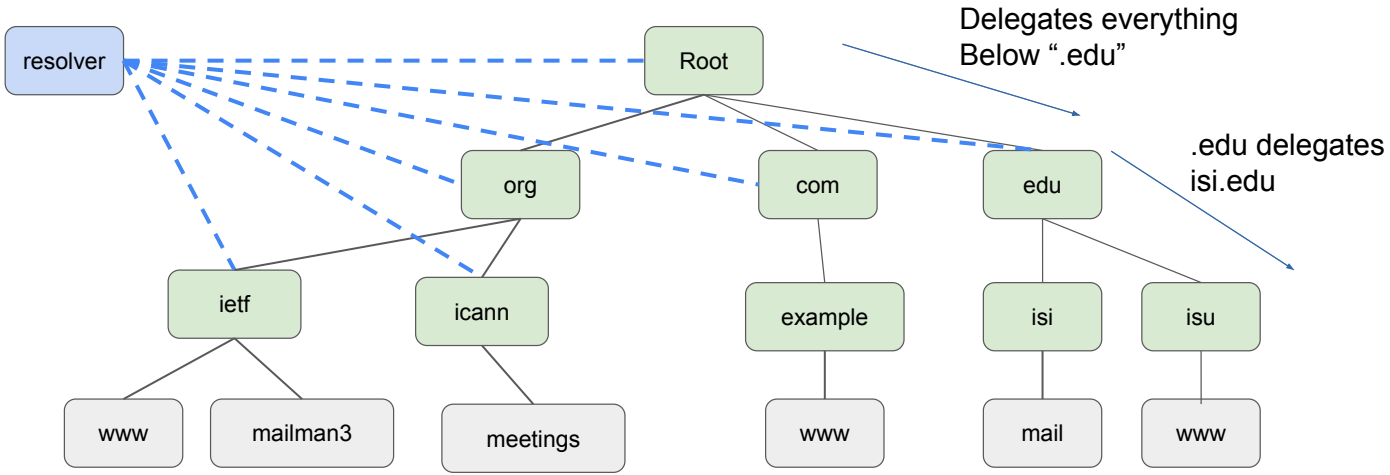
DNS Resolvers Communicate With Everything



Greater visibility of DNS requests and responses

Reminder: this requires a machine-in-the-middle already!

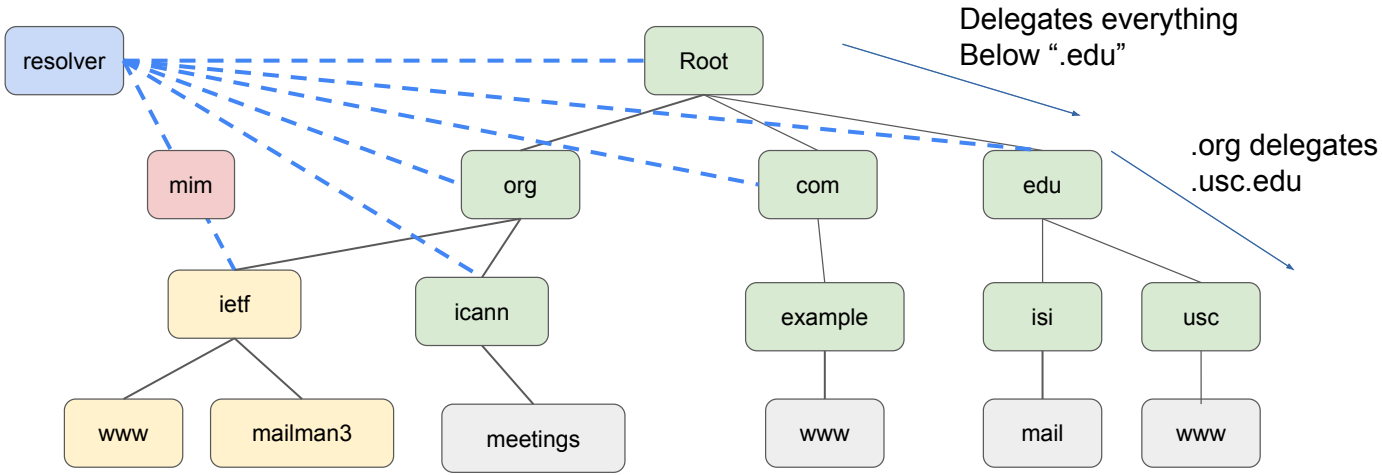
But: I can now force my vantage point in front of others



Greater visibility of DNS requests and responses

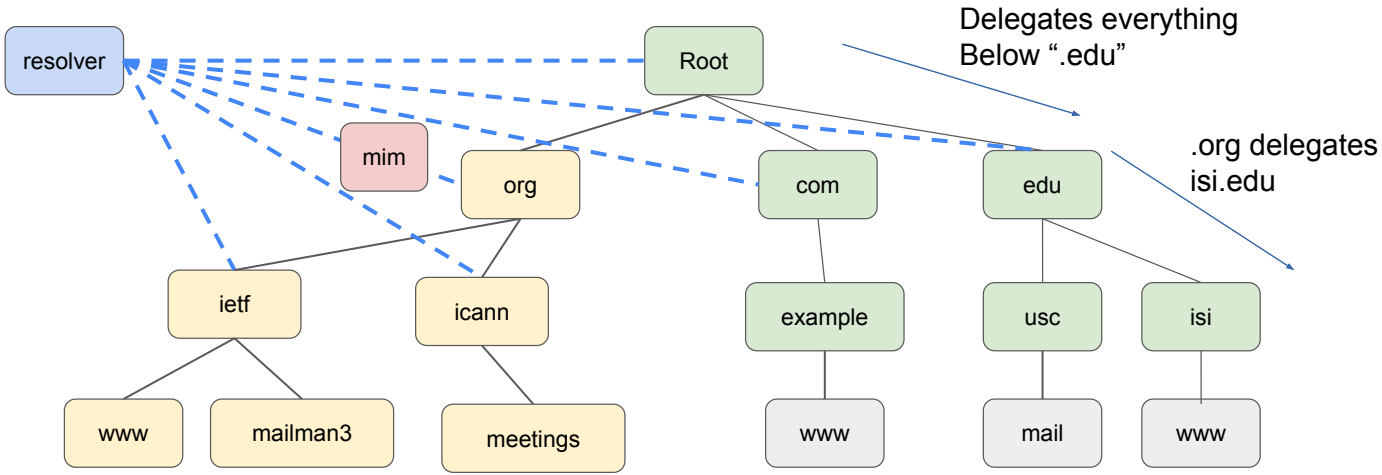
Reminder: this requires a machine-in-the-middle already!

If the MiM can only see one domain's traffic…. not a big deal.



Essentially: complete visibility for everything below a point

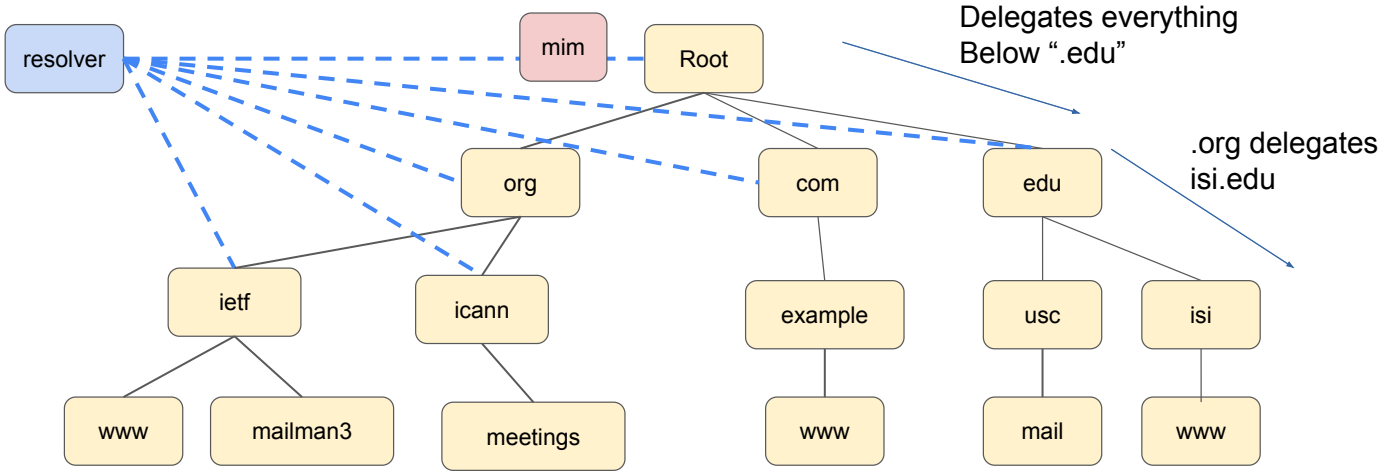
If the MiM can only see a middle point in the tree, they can see everything below



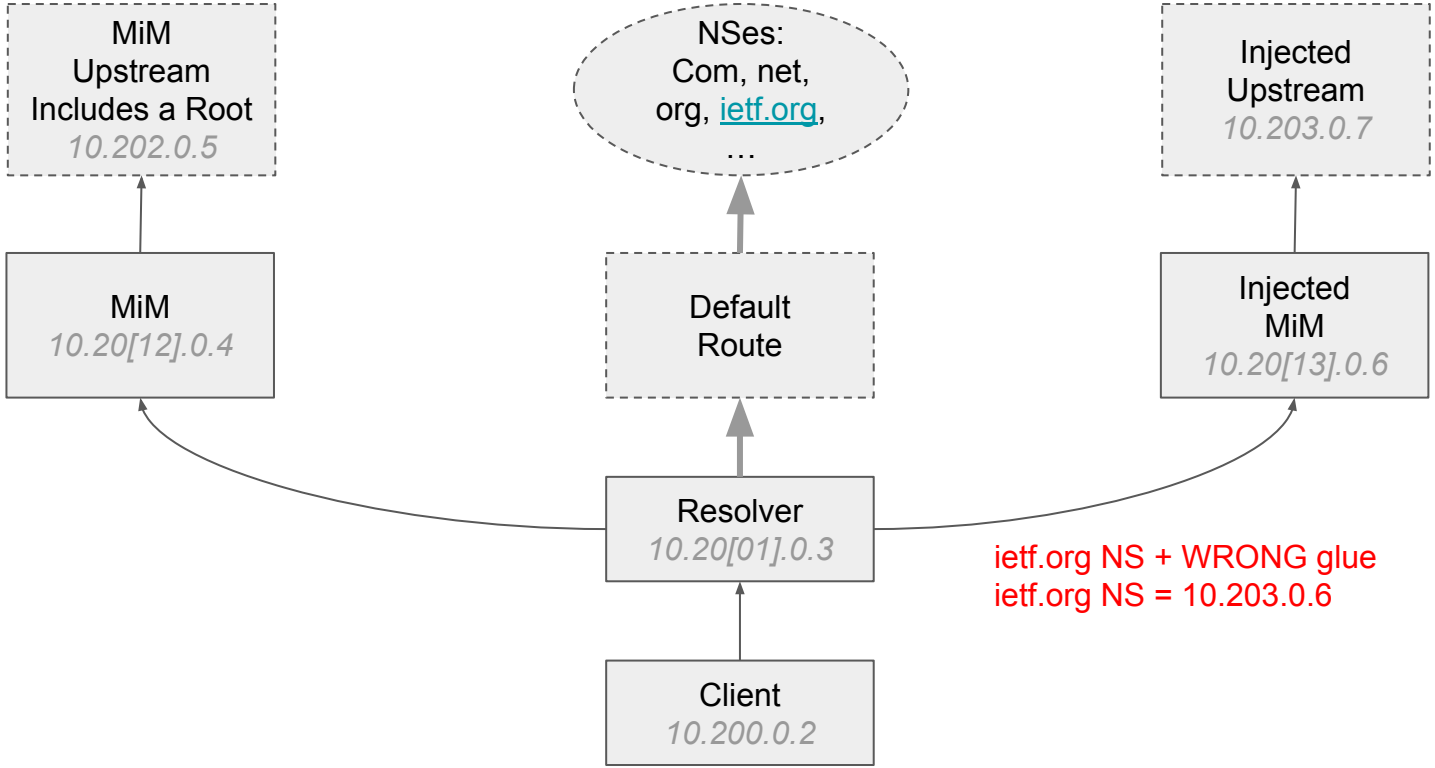
Greater visibility of DNS requests and responses

And a MiM before the root gives a vantage point of the entire Internet's DNS

Even though **ALL** these zones are **DNSSEC signed**



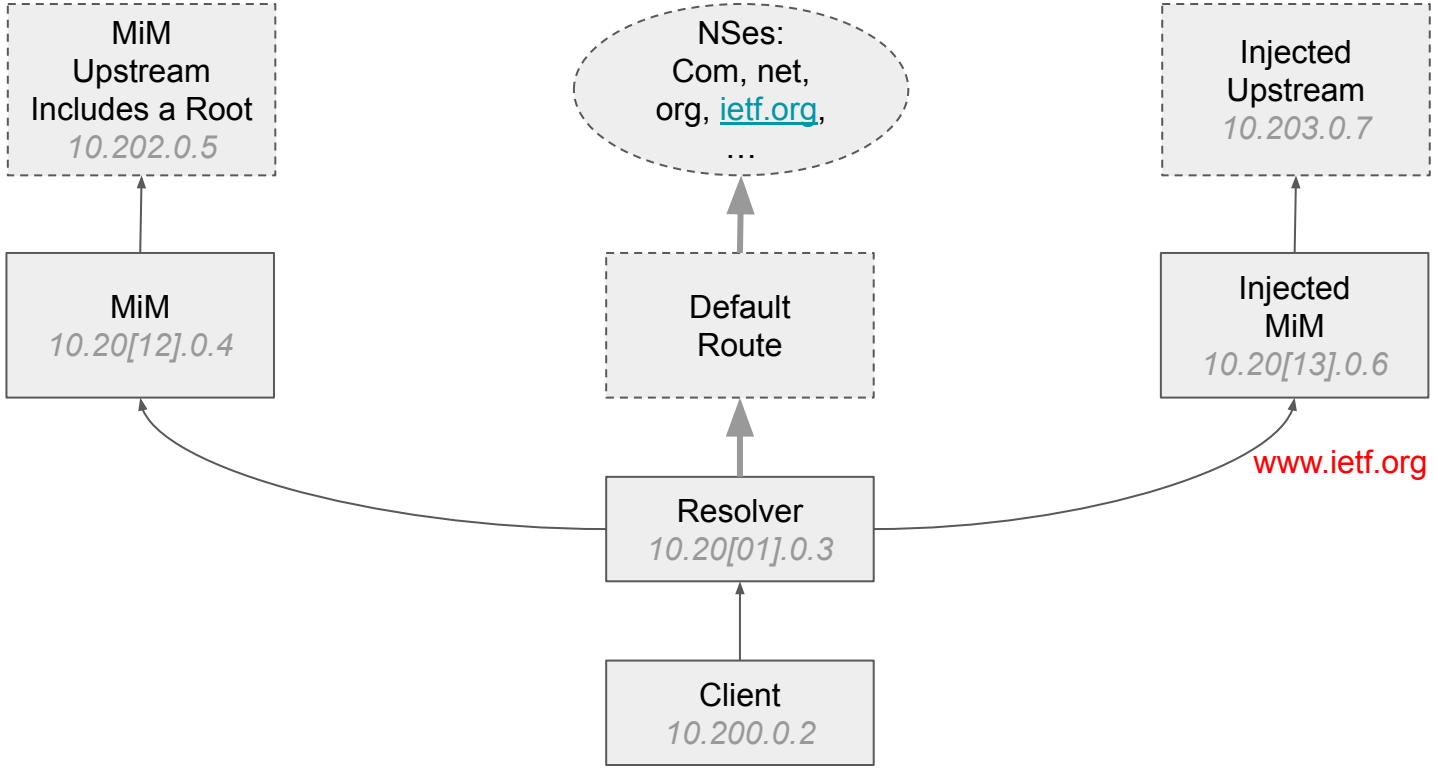
Test Network Design



Parent Cache
.org/NS = 10.203.0.6
ietf.org/NS = 10.203.0.6

Child Cache

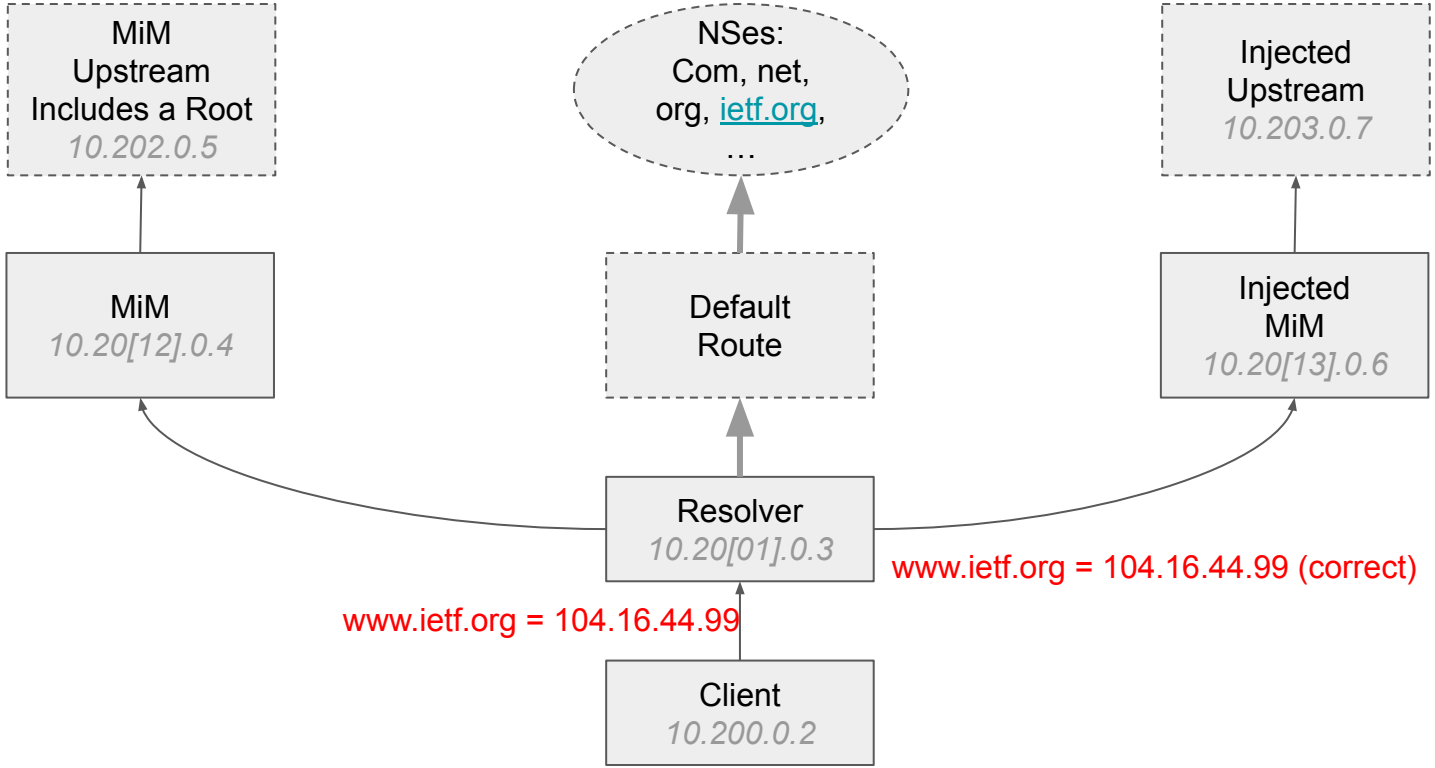
Test Network Design



Parent Cache
.org/NS = 10.203.0.6
ietf.org/NS = 10.203.0.6

Child Cache

Test Network Design



Parent Cache
.org/NS = 10.203.0.6
ietf.org/NS = 10.203.0.6

Child Cache
www.ietf.org = 104.16.44.99

*** Important demonstration components

1. Resolver routes are configured so all root traffic goes left
    a. All other Internet traffic goes out the default
2. The MiM cannot forward traffic to the root without modification
3. Only traffic to the root must come from the MiM source address
4. The MiM intercepts packets and:
    a. Modifies the source address to be its own
    b. Sends requests to an upstream resolver
    c. Gets the responses back and modifies any glue to its own IP address (or the injected MiM)
    d. If the type = NS, modifies all A responses to its own IP address
    e. Replaces the source / destination addresses to match the original request
    f. Delivers it back to the resolver under attack

Demonstration Modes

A. Normal routed traffic – only the root traffic is seen by the MiM
B. MiM injecting itself in front of all DNS requests
C. MiM injecting another MiM in front of all requests
    a. Which subsequently injects itself for all other requests