



# Getting started with EDNS Client Subnet

Cathy Almond

26 October 2023

<https://www.isc.org>



# What this webinar will cover:

- What (and why) ECS?
- Simple first configuration
- Prefixes and scopes
- ECS forwarding
- ECS cache
- “Gotchas” and mitigations

# What's the problem?

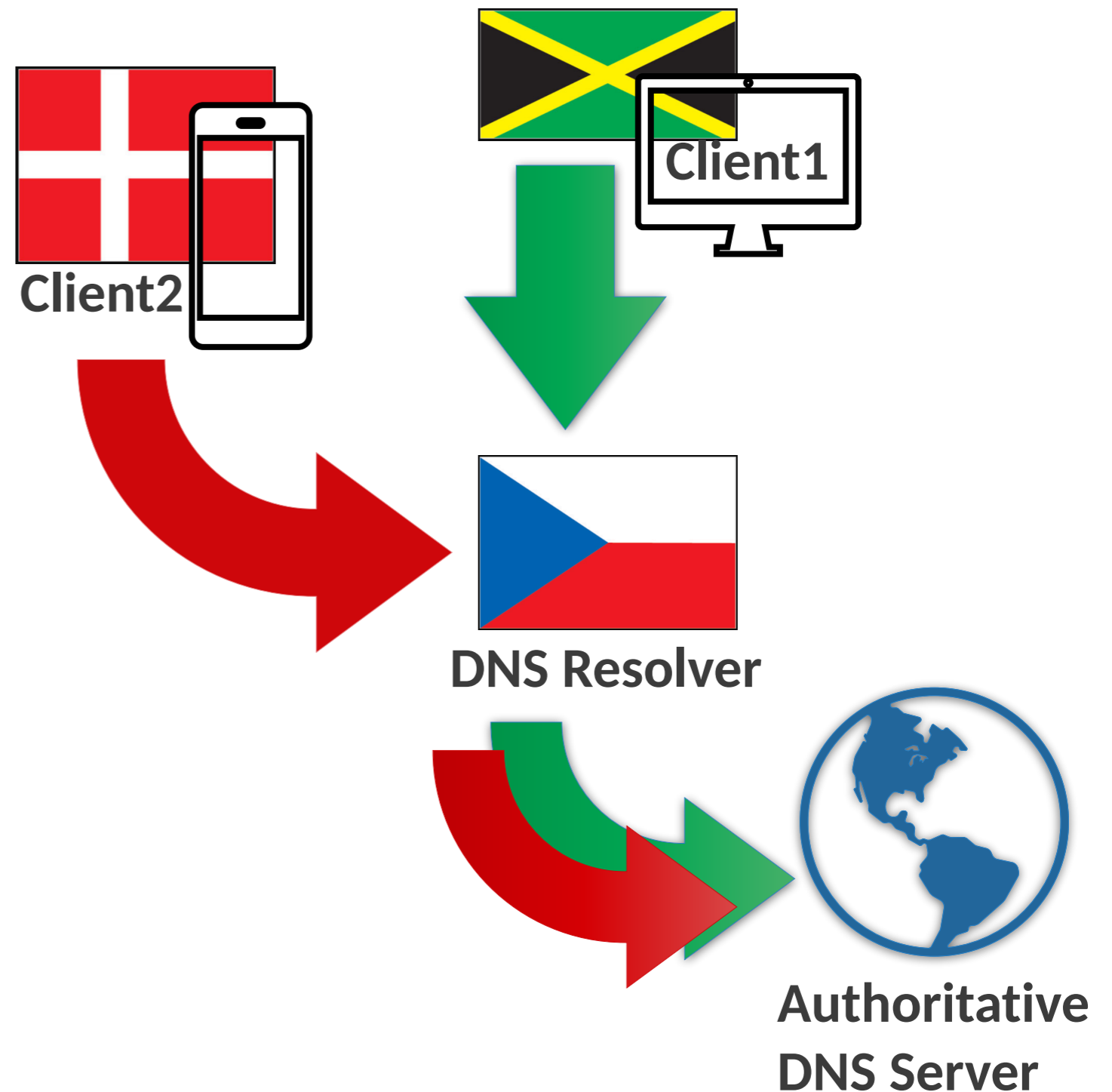
- Some authoritative servers provide location-specific answers
- Resolver location may not represent client location!



Photo by [NASA](#) on [Unsplash](#)

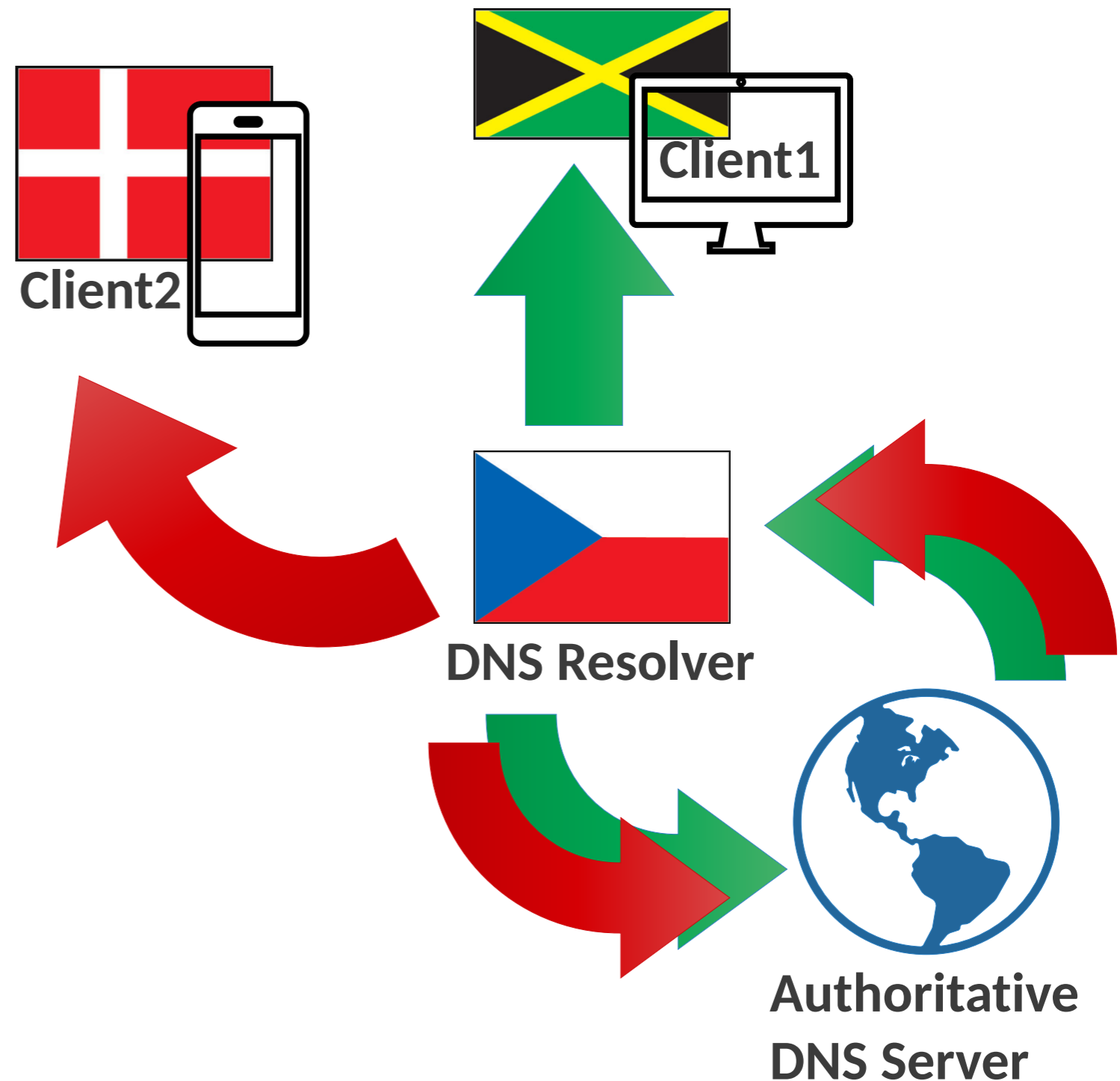
## What if ...

- A resolver making a query could 'tell' the authoritative server where the client it is serving, is located?



## What if ...

- Authoritative servers returned different answers to resolvers who told them where their clients are located?



# EDNS Client Subnet (ECS)

- ECS-enabled Resolver can add client subnet to queries it sends to authoritative zone servers
- ECS-supporting authoritative servers reply, adding subnet and mask (scope) to their query responses
- Resolver caches answers **with** ECS scopes

<https://datatracker.ietf.org/doc/html/rfc7871>



# Let's try it!

```
$ dig @216.239.32.10 +qr +noredc google.com +subnet=82.71.29.0/24

; <<>> DiG 9.16.42-S1 <<>> @216.239.32.10 +qr +noredc google.com
+subnet=82.71.29.0/24
; (1 server found)
;; global options: +cmd
;; Sending:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 56625
;; flags: ad; QUERY: 1, ANSWER: 0, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; CLIENT-SUBNET: 82.71.29.0/24/0
; COOKIE: 8418eadccb28cf24
;; QUESTION SECTION:
;google.com.          IN A

;; QUERY SIZE: 62
```



# Oooh!

```
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 56625
;; flags: qr aa; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 512
; CLIENT-SUBNET: 82.71.29.0/24/18
;; QUESTION SECTION:
;google.com.          IN A

;; ANSWER SECTION:
google.com.  300  IN A  142.250.180.14

;; Query time: 63 msec
;; SERVER: 216.239.32.10#53(216.239.32.10)
;; WHEN: Wed Oct 25 13:34:32 BST 2023
;; MSG SIZE rcvd: 66
```





# And also with IPv6

```
$ dig @216.239.32.10 +qr +norec google.com +subnet=2001:500:6b::/56

; <<>> DiG 9.16.42-S1 <<>> @216.239.32.10 +qr +norec google.com
+subnet=2001:500:6b::/56
; (1 server found)
;; global options: +cmd
;; Sending:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 5053
;; flags: ad; QUERY: 1, ANSWER: 0, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; CLIENT-SUBNET: 2001:500:6b::/56/0
; COOKIE: 81c7f1f55530bb43
;; QUESTION SECTION:
;google.com.          IN A

;; QUERY SIZE: 66
```



# And also with IPv6

```
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 5053
;; flags: qr aa; QUERY: 1, ANSWER: 6, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 512
; CLIENT-SUBNET: 2001:500:6b::/56/48
;; QUESTION SECTION:
;google.com.      IN A

;; ANSWER SECTION:
google.com.      300  IN A  74.125.138.139
google.com.      300  IN A  74.125.138.102
google.com.      300  IN A  74.125.138.138
google.com.      300  IN A  74.125.138.113
google.com.      300  IN A  74.125.138.101
google.com.      300  IN A  74.125.138.100

;; Query time: 49 msec
;; SERVER: 216.239.32.10#53(216.239.32.10)
;; WHEN: Thu Oct 26 11:33:59 BST 2023
;; MSG SIZE rcvd: 150
```



# But...

- ECS has not been standardized by the IETF. There is concern about the privacy implication of providing additional information about the client to unrelated Internet authorities
- ECS should be deployed only if the benefit outweighs the cost
- Are you doing DNSSEC validation and will it still work?
- Once you enable ECS in your BIND 9 resolver, the resolver will cache the different responses for each subnet, significantly inflating cache size
- Not all authoritative servers properly handle queries containing ECS options

# ECS in BIND Subscriber (-S) Edition

- BIND9 -S edition includes configurable resolver ECS options
- BIND does not include an authoritative ECS solution

<https://www.isc.org/docs/BIND-9-S-Edition.pdf>



# Getting started...

BIND9 -S edition  
“out of the box”  
does not  
automatically do *any*  
ECS - it behaves in  
exactly the same  
way as Open  
Source BIND



Photo by [krakenimages](#) on [Unsplash](#)

# Getting started ...

- The first (and possibly only) option you need is: **ecs-zones**
  - Use this configuration option to list the domains to whose servers you wish to send ECS options
  - You can also use **ecs-zones** to exclude some domains and domain names

# ecs-zones

- Enabling resolver ECS for queries to *all* authoritative servers:

```
ecs-zones { . ; } ;
```

- Enabling resolver ECS for a specific list of domains only:

```
ecs-zones { example.com ;  
example.net ; } ;
```

## ecs-zones

- A more complex ecs-zones example:

```
ecs-zones {example.com;  
example.net;  
!excluded.example.com;  
!excluded.example.net; };
```



## But oops?...

- The domain names specified in **ecs-zones** are matched against the **domain whose servers are being queried by your resolver**, not against the client query name
- **ecs-zones** is *effectively* specifying servers (by listing the domain for which they are authoritative); *it is not a filter being applied to client queries themselves*



Photo by [Nathan Dumlao](#) on [Unsplash](#)

# Example

- Client query is "fluffy.cats.example.com"
- named.conf has:

```
ecs-zones {cats.example.com;};
```
- Resolver sends query to (already learned) servers for [example.com](#)
- These servers reply authoritatively with an answer for "fluffy.cats.example.com"
- Resolver will *not* be using ECS for this query because it has queried (and received a reply) from the servers it knows as authoritative for example.com, and never queries any servers for cats.example.com.

# ecs-zones

- Listed zones match the authority of the servers being queried:

```
ecs-zones {example.com;  
example.net;  
!excluded.example.com. ;  
!excluded.example.net; };
```

- **However, listed exclusions are handled differently - these instead match the client query name**
- In both instances we're doing closest match, thus queries for **this.is.my.example.com** will use ECS and those for **really.excluded.example.net** will not

# Example

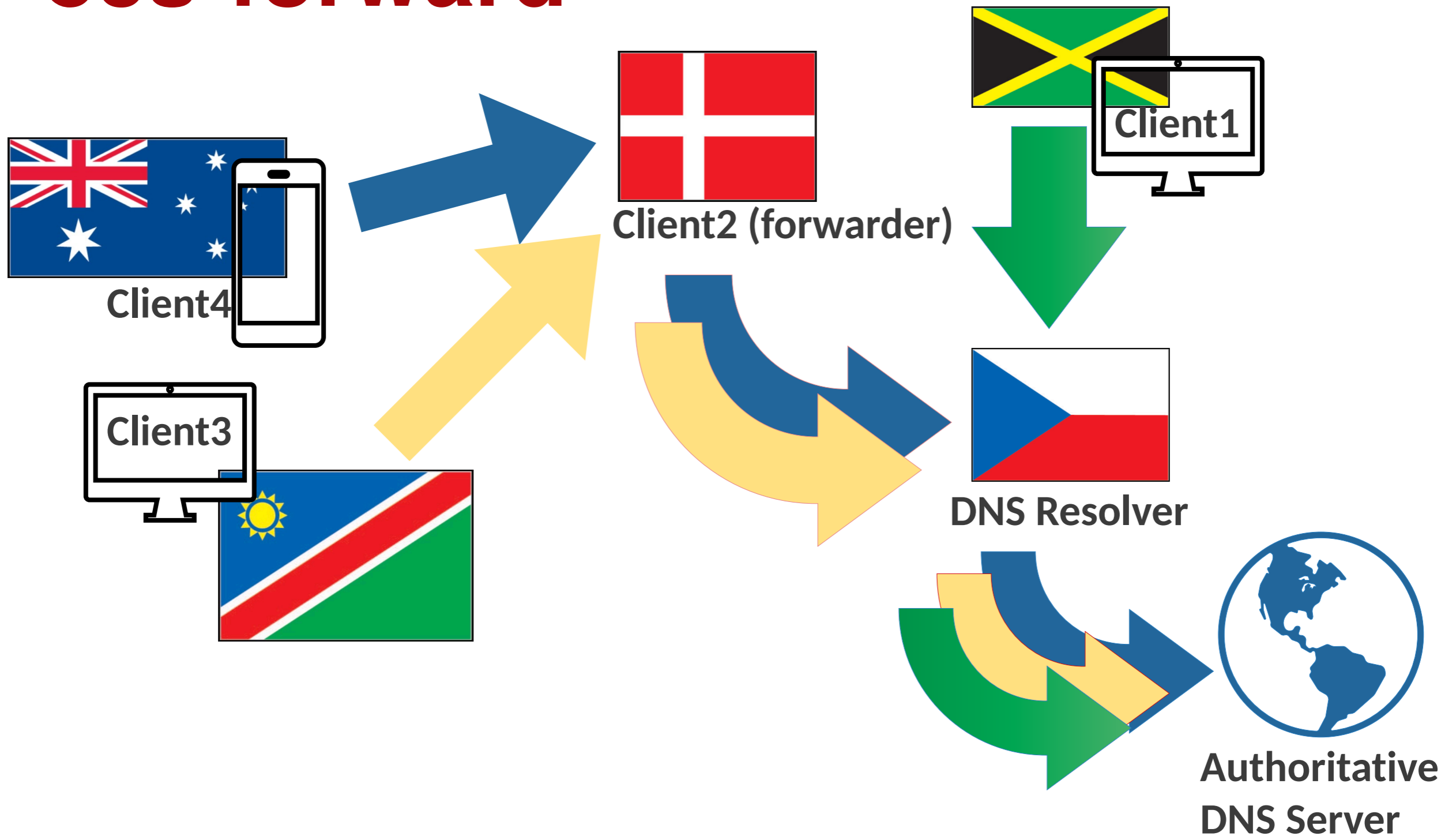
- Client query is "spotty.cats.example.com"
- named.conf has:

```
ecs-zones {example.com;  
!spotty.cats.example.com};
```
- Resolver sends query to (already learned) servers for [example.com](#) (*we fixed named.conf option ecs-zones from the earlier example/test so that we would now use ECS*)
- Resolver will *not* be using ECS for this query because although it has matched the destination servers for [example.com](#), we've explicitly excluded name [spotty.cats.example.com](#)

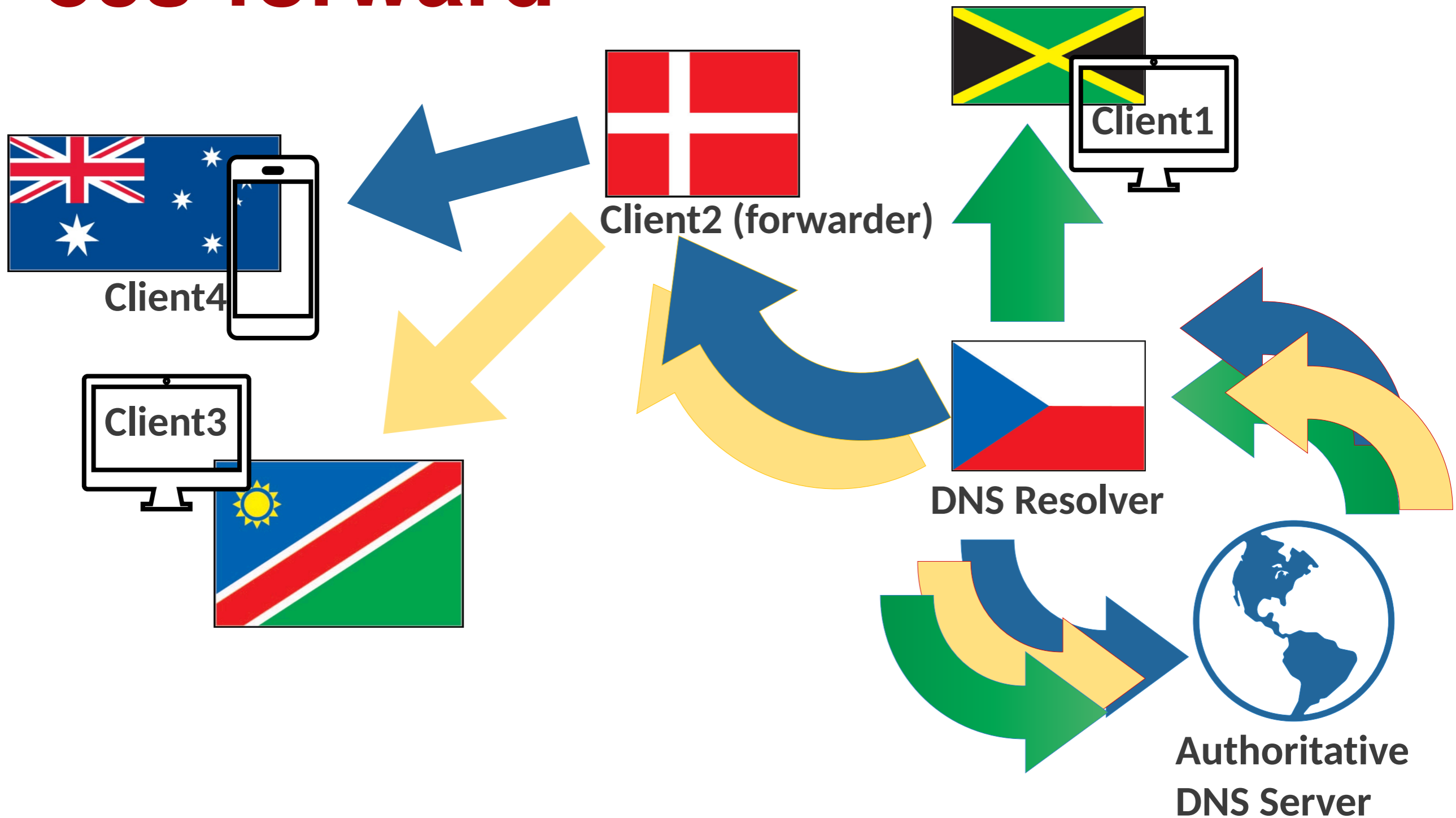
# Becoming more sophisticated ...

- There are other ECS options available:
  - Use **ecs-forward** to permit clients to send their own ECS options, which may then be forwarded by the resolver
  - Use **ignore-ecs-opt** to discard unsolicited ECS in server query responses
  - Use **ecs-types** to restrict the RR types that are eligible for ECS

# ecs-forward




# ecs-forward





# ecs-forward

- Option **ecs-forward** specifies an ACL of client addresses from which ECS-tagged queries may be forwarded
- If the client sending the query is allowed, and the query name would be allowed per **ecs-zones**, then the resolver ECS processing uses the received client option (*with some provisos - see **ecs-bits** later in this presentation*)
- A client query received with global scope ECS option (prefix 0) effectively disables ECS for this query
-  Client queries containing **non-global** ECS options where the client is not included in **ecs-forward** will be **REFUSED**



# ecs-forward

- Enabling forwarding of ECS options for specific clients and subnets:

```
ecs-forward {  
192.0.2/24;  
2001:db8::/32;  
};
```



# ignore-ecs-opt

- Open Source BIND is unaware of ECS and will ignore unsolicited ECS options in query responses from authoritative servers.
- BIND9 -S edition with no ECS configuration at all, will do the same
- BIND9 -S edition with any ECS configuration will drop/reject any query responses from authoritative servers that contain unexpected ECS options
- Use **ignore-ecs-opt yes;** to ignore 'surprise' ECS options from broken DNS server implementations

# ecs-types

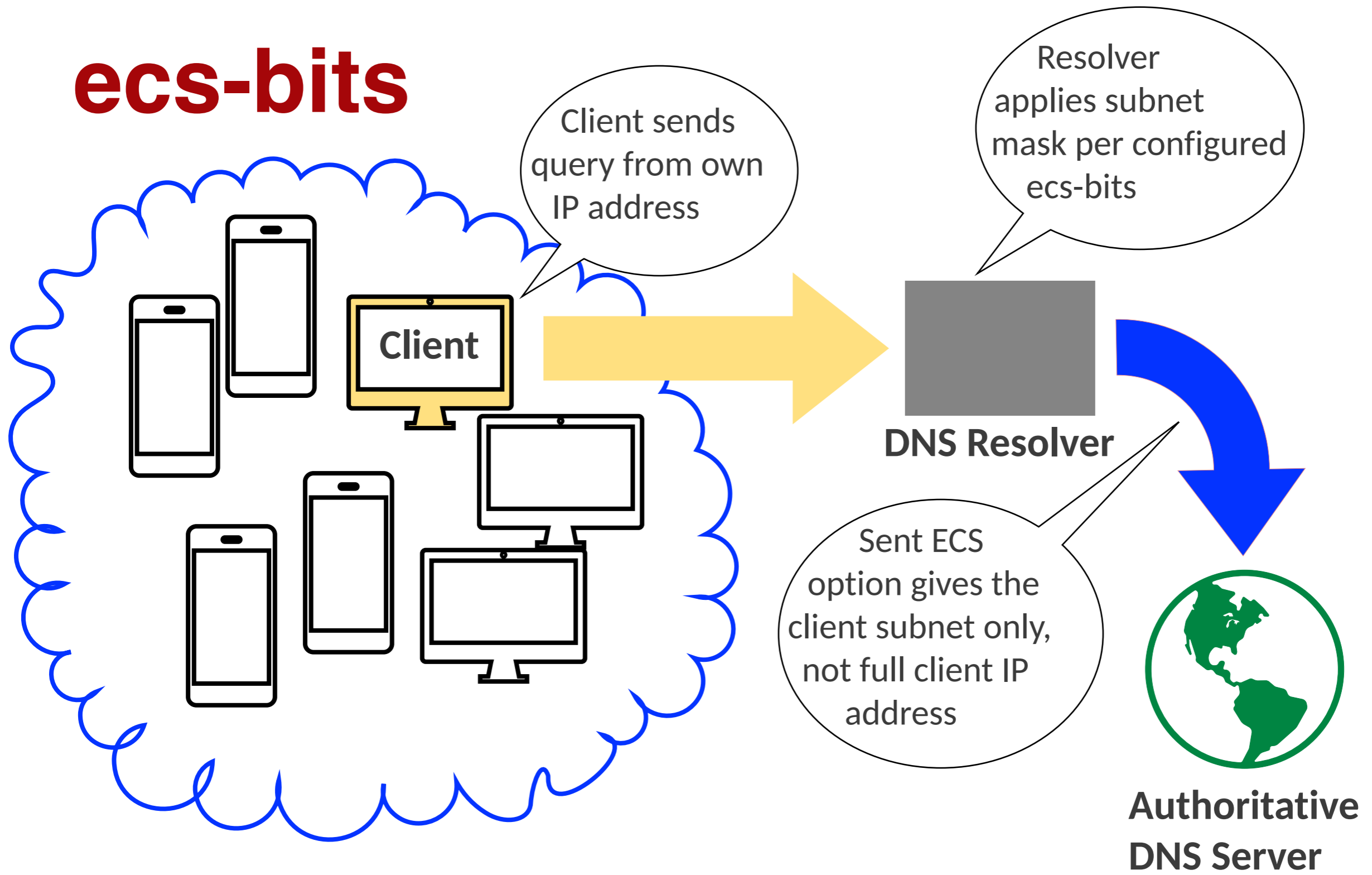
- By default, when using ECS for a query, the resolver does so for all RR types except DNS infrastructure (NS and SOA) and some DNSSEC types
- You can instead (if you wish) use option **ecs-types** to list which RR types being queried will use ECS
- CNAME is special - even if not listed in **ecs-types**, if the original query type is listed, an ECS-scoped CNAME query response RR will still be used and cached



# Privacy concerns

- There are several ECS options available:
  - Restrict the granularity of advertised ECS subnet information using global option **ecs-bits**
  - Use **bits-v4** and **bits-v6** within **ecs-zones** for per-zone ECS granularity
  - Use **ecs-privacy** to signal upstream that ECS is to be disabled
  - Server option **ecs no**; will prevent ECS options being sent to specific servers or address ranges

# ecs-bits





# ecs-bits

- Provides the default prefix length (subnet mask) to use in ECS queries for IPv4 and IPv6 addresses
- Default values are 24 (for IPv4) and 56 (for IPv6)
- These are also the maximum allowed (unless overridden when building BIND, using compiler flags
  - DECS\_MAX\_V4\_SCOPE and/or
  - DECS\_MAX\_V6\_SCOPE
- Changing the maximum does not also change the default!
- *Current packaged versions of BIND-S edition are built with -DECS\_MAX\_V6\_SCOPE=64*



# ecs-bits

- Configuring the defaults explicitly:

```
ecs-bits 24 56;
```

- More client privacy:

```
ecs-bits 16 48;
```

- No client privacy at all (perhaps for an internal-only service?):

```
ecs-bits 32 128;
```

## ecs-zones options bits-v4 and bits-v6

- It is possible to override the default source (requested) prefixes on a per-zone basis from the **ecs-zones** option
- As names in **ecs-zones** become more specific, prefix-lengths cannot increase. If **example.com** is specified with **bits-v4 20**, then no prefix length higher than 20 can be used for IPv4 queries for any subdomain of **example.com**



# Example

```
ecs-zones {  
example.com bits-v4 20;  
! excluded.example.org;  
example.org bits-v4 22 bits-  
v6 48;  
example.net;  
};
```



# ecs-bits vs. ecs-forward

- If you are using **ecs-forward** and a client query has a requested scope with a larger prefix than allowed for this zone (as controlled by **ecs-bits** and **bits-v4/bits-v6**), then the resolver will adjust the client options to truncate the subnet before forwarding
- If you are using **ecs-forward** and a client query has a requested scope with a smaller prefix than allowed, the resolver uses the client options
- Any client can disable ECS using a zero length prefix (even if not listed in **ecs-forward**)



# ecs-privacy

- If set to yes, then when a query is allowed for ECS processing and no client ECS option is being forwarded, the resolver will always include an ECS option with a source prefix-length of zero in all of its upstream queries.
- This is a request to upstream intermediate resolvers to disable ECS when processing queries sent by this resolver.
- The default is no.
- You would potentially use this on a resolver that handles only forwarded client queries, so that it doesn't ever add its own ECS options if the client forwarders didn't request them



# Server option “ecs no;”

- For when you need to override the sending of ecs options on a per-server basis...



# ECS cache

- Ordinary cache
- ECS cache - extra RRsets maintained in cache alongside ordinary records, with the scope that the authoritative server provided
- Global scope - RRsets held in ECS cache with prefix 0 - these will be used to match all client queries where ECS is allowed (per **ecs-zones**)
- *Caches for resolvers using ECS are going to be larger!*



# ECS cache

- Authoritative servers provide answers at the requested scope, or with a smaller prefix
- Authoritative servers that *could* provide answers with a larger prefix, may indicate this by returning a larger prefix value in their response
- Negative authoritative server responses are all cached with global scope
- Delegation responses are also cached with global scope

# ECS cache

- Use `rndc dumpdb -ecscache` to dump cache content, including ECS-scoped RRsets
- Use `dig` option `+subnet=` to test resolver behaviour
- Use `dig` options `@<server IP> +norec +subnet=` to test authoritative server responses
- Use `dig` options `@<server IP> +norec +subnet=` to find out what an ECS-enabled resolver has in cache



# “Gotchas”

- Cache bloat
- DNSSEC-validation
- Unsolicited ECS options on server responses
- Unexpected client-supplied ECS options
- Negative server responses are globally cached
- Transient CNAMEs used by CDNs





# Also we have heard that ...

- Some authoritative servers “break” when receiving queries with ECS options
- Other authoritative servers always respond with the same answer RR but scoped to match the ECS subnet requested - unnecessary cache bloat!
- Surprise unsolicited ECS options on server responses
- GeoIP databases are not always correct
- Authoritative zone providers may use different GeoIP databases for sender IP (if no ECS) and subnet specified in ECS options

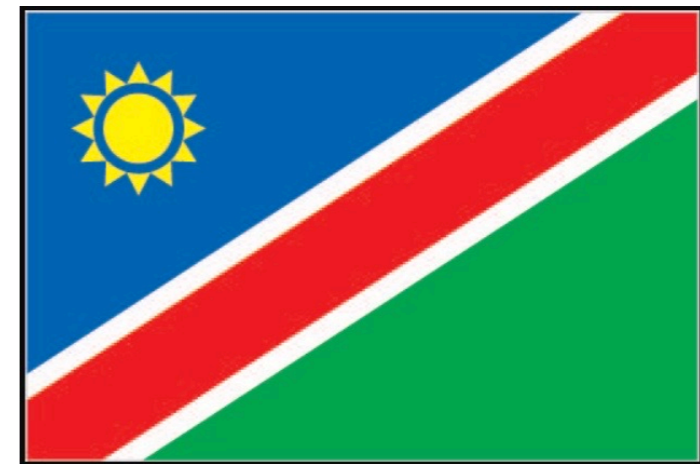
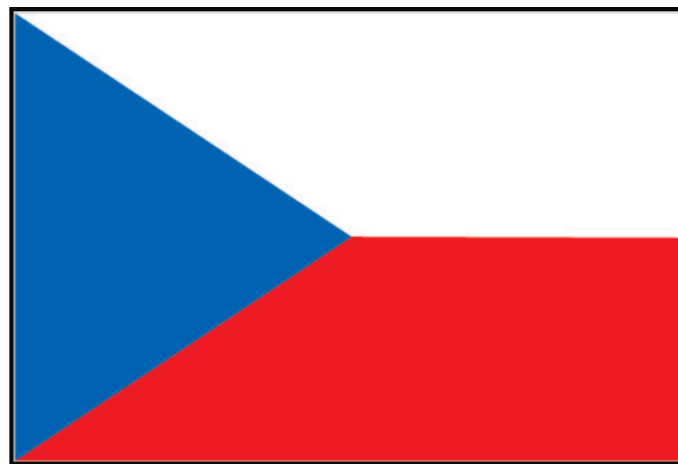
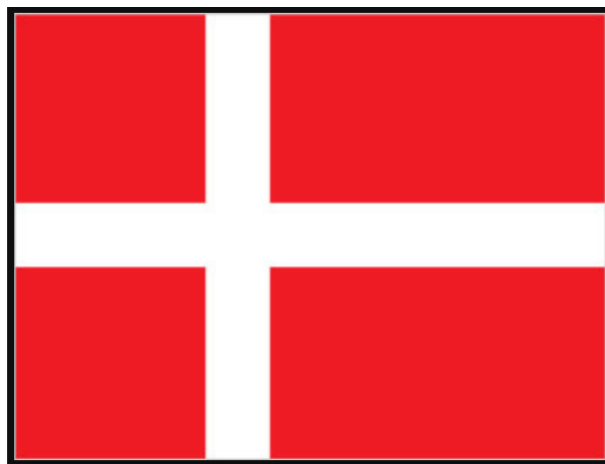
# Summary of topics covered:

- What (and why) ECS, and effective use of:
  - **ecs-zones** (and optional **ecs-types**)
  - **ecs-bits** (also per-zone **bits-v4**, **bits-v6**)
  - **ecs-forward** (and **ecs-privacy**)
  - server options **ecs** and **ignore-ecs-opt**
- ECS cache
- Problems and mitigations



# Any Questions?

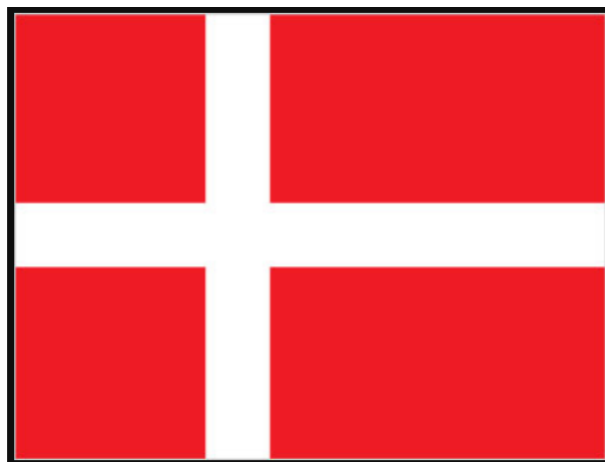
**(No prizes) quiz - identify the flags:**



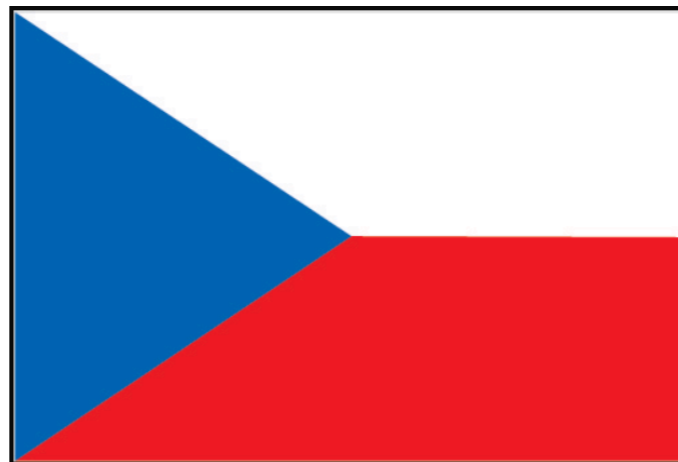
# Thank you!

- ECS RFC: <https://datatracker.ietf.org/doc/html/rfc7871>
- ECS KB Article: <https://kb.isc.org/docs/edns-client-subnet-ecs-for-resolver-operators-getting-started>
- Main website: <https://www.isc.org>
- Software downloads: <https://www.isc.org/download> or <https://downloads.isc.org>
- Presentations: <https://www.isc.org/presentations>
- Main GitLab: <https://gitlab.isc.org>

# Flags:



Denmark



Czechia



Namibia



Jamaica



Australia