Networks: DNS

i.g.batten@bham.ac.uk

4693160**0**

Ouch!

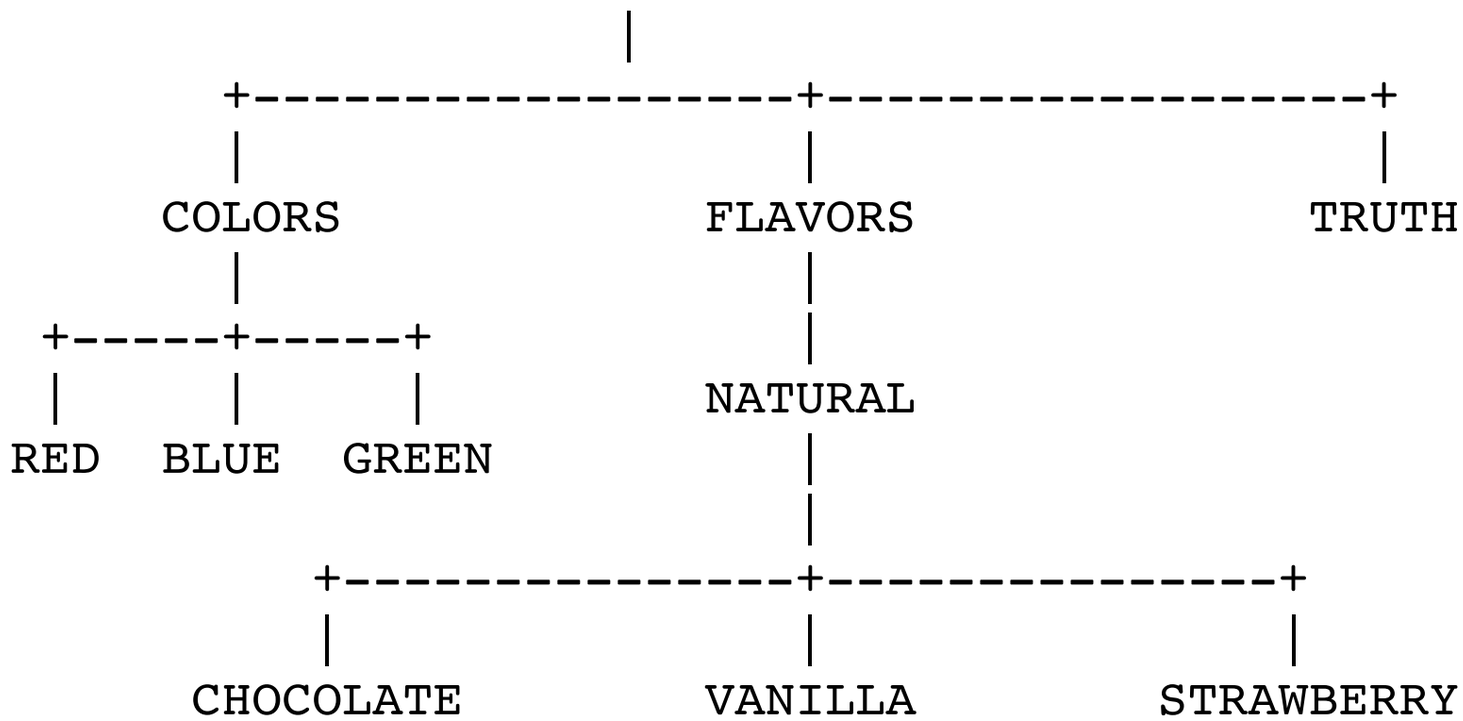
469316**00**

What is DNS?

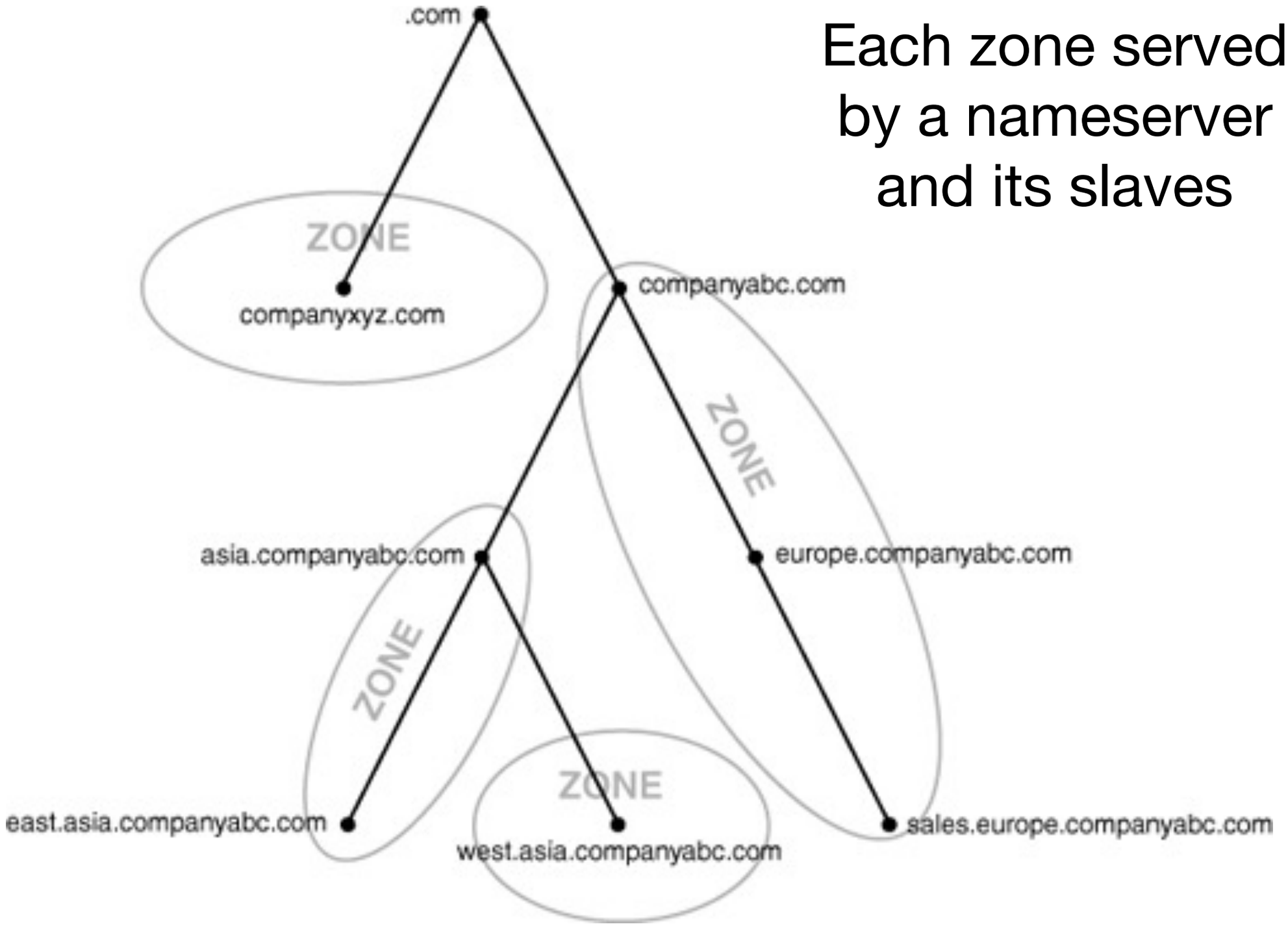
- Absolutely fundamental to Internet: maps names to IP numbers (v4 and v6), numbers to names, locates resources
- One of the oldest protocols in regular use (RFC 882, November 1983, current protocols pretty much as they now are in RFC1034/RFC1035, November 1987)
- Hideously insecure, complex implementations, politically sensitive and a nightmare for governance

DNS Concept

From RFC882



Zones



Each zone served by a nameserver and its slaves

Outline

- Everyone that needs one can get a domain name and operate a nameserver at that point in the hierarchy.
- Once they have that domain name, they can create **resource records** within the domain, to an arbitrary depth
  - Not quite: DNS names are limited to 255 bytes, 63 bytes per label
- They can also **delegate** portions of the namespace to other nameservers
- A group of resource records with a common root served from one primary nameserver is called a **zone**

DNS as Database

- The DNS forms a loosely coupled distributed database, containing key/value pairs
  - Sometimes grossly abused for that purpose, as we will see later
- Lots of caching and redundancy, dating back to a slower, less reliable, less connected Internet
- As a general rule, everyone's DNS infrastructure is broken, and the definition of "not broken" is the topic of much debate
  - I hope I'm treading a middle-of-the-road position

DNS Components

- **Clients**: ask questions to **recursive servers** and receive answers
- **Recursive Servers:** (sometimes called **caching servers**) can be asked a complete question (“what is the address of some.machine.dom.ain?”) and will give a complete, but sometimes **inauthoritative** answer — they do the heavy lifting
- **Authoritative Servers**: (sometimes called **iterative servers**) only give **authoritative** answers about zones they are configured to know about.
  - **Primaries** and **Secondaries** are both **Authoritative**

Resource Records (RR Sets)

- Map a name to some data, plus have some book-keeping data in them.
- Simple case, A records contain IPv4 addresses
- AAAA records contain IPv6 addresses

Name	TTL	Class	Type	Data
gromit.cs.bham.ac.uk.	86400	IN	A	147.188.193.16
research-1.batten.eu.org.	86400	IN	AAAA	2001:630:c2:3263:8:20ff:fe89:b5a0

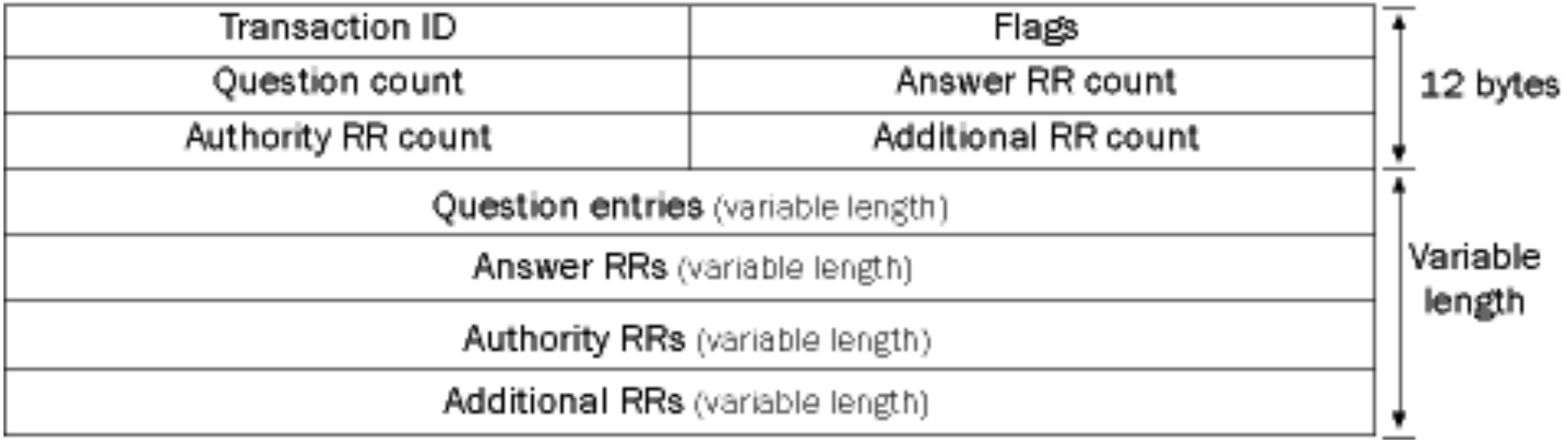
RR Sets

- Class is always IN for the INternet (older classes no longer relevant, wise implementors reject them out of hand)
- Types include A, AAAA, PTR (address->name), MX (mail exchangers), NS (nameservers), CNAME (aliases), SOA (authority records) and TXT (random dumping ground for textual information)
- Can be multiple records for a given name, hence RR **sets**.

TTL

- TTL: “Time to Live”, usually in seconds
  - You quickly learn that 3600 is an hour, 86400 is a day, 604800 is a week.
- You can cache this resource record for this long

Packets



16 bit transaction ID: serious problem for security in noughties (“Kaminsky Attack”), still relevant today under some realistic conditions

Flags

- Bit 0: QR – 0 is Question, 1 is Response
- Bits 1–4: OPCODE (0 query, 1 inverse query, 2 status, 1 and 2 optional, rare, ill-defined and essentially unused)
- Bits 5–8: AA (answer is authoritative), TC (this answer has been truncated, please try TCP), RD (recursion desired), RA (recursion available)
- Bits 9–11: reserved
- Bits 12-15: RCODE (0 OK, 1 Format error, 2 Server error, 3 Name error, 4 Unimplemented, 5 Refused)

RRSETs

- Header says how many of each type there are
- Body contains that many
- Questions: the query I want an answer to (or for which this is an answer)
- Answers: response to the query
- Authority records: where is the origin for this data?
- Additional records: any stuff the server has to hand which the client might find useful (addresses for NS or MX, in particular)

Clients

- An utter shambles on most operating systems
- DNS clients are also known as “resolvers”
- You can make direct queries to the DNS using the facilities of libresolv (res_mkquery)
- Normally you call getaddrinfo() or gethostbyname() and that chooses mechanism from DNS, local files, LDAP, NIS/YP (is Ronald Reagan still president?) and so on
  - /etc/nsswitch.conf is common, originally Ultrix (/etc/svc.conf), then Solaris, now Linux as well — maps queries (hosts, passwd, printers) to sources (DNS, files, NetInfo, whatever)
- Modern systems maintain a cache over all this (nscd on most Unixes).
  - nscd caches tend to be sloppy with TTL handling and cache everything for up to an hour.

Recursive Servers

- Know how to answer a complete question (mechanism to follow)
- Should be access-control and firewall restricted to local network
- Once they have resolved a name they can cache the result, and can answer repeated questions from the cache so long as they appropriately decrement the TTL

Authoritative Servers

- For a given domain, there will be one or more nameservers specified in some zone logically closer to the root (not necessarily one node up).
- Each of these specified nameservers is **authoritative** for the specified domain.
- The NS record that points to them is called a **delegation** in the **parent.**

Authoritative Servers

- Why multiple servers?
  - Load balancing and redundancy
  - Ideally, spread over the world, spread over multiple ASes
    - Anycasting is useful: a domain with just one NS record may be using it
- They are all equally authoritative: no concept of “master” or “slave” is exposed
- They hold “zone files” for the zone files they are authoritative for
  - Either as real files, or as “zones” within the server software

How are names resolved?

- Clients ask recursive servers (“rd” = “recursion desired”)
- Recursive servers start at the root, and **iterate** downwards asking for the nameserver of the next label down, until they can finally ask the last nameserver for the required RR

Flow of packets

- Recursive nameserver is pre-configured with addresses of nameservers for “.”, the root of the domain space.
- To answer a query for “A foo.dom.ain”, asks one of the root nameservers “NS? foo.dom.ain”.
- Will get back either the location of the nameservers for “ain”, or “dom.ain”, or “foo.dom.ain”, depending on what knowledge the server has.
- Can then ask next server down the same question, until someone answers with an A record.
    - Called “iteration”, although actually feels somewhat recursive.

Resource Records

- Name TTL Class Type Data
- foo.domain.mytld is represented as [3] foo [6] domain [5] mytld [0] where [3] is a byte with the value 3, 0x3, 00000011.
- Labels can be compressed

Label Compression

- Labels are maximum 63 bytes, so largest value for a length field is 00111111.
- Length fields starting 11 (ie, >= 192) are special. If length field starts 11 (ie length & C0 == C0) then next six bits plus the next byte, total fourteen bits, are special.
- 14 bits are read as “read from this offset in message to next zero”.

Compression

- Suppose [3] foo [3] dom [3] ain [0] starts at byte 48 in the message.
- Suppose we want to represent the name mail.foo.dom.ain.
- Choice 1: [4] mail [3] foo [3] dom [3] ain [0], 18 bytes.
- Choice 2: [4] mail [192] [48], 7 bytes
- Particularly effective as queries about things in domain x.y.z tend to have a lot of x.y.z in them.
- Compression goes to end of name, you can’t “reuse” labels like www and mail from other domains.

Caching

- At each stage, everyone caches the data they get
- Particularly, the recursive server will rapidly build a cache of the NS records for popular domains, only refreshed once per TTL seconds
- In original protocol, anyone can supply cached information about anything and be believed
- Modern systems are more sceptical

Caching In Action

```
ians-macbook-air:clocks igb$ dig aaaa rsync.batten.eu.org

; <<>> DiG 9.8.3-P1 <<>> aaaa rsync.batten.eu.org
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 15509
;; flags: qr aa ra; QUERY: 1, ANSWER: 1, AUTHORITY: 7, ADDITIONAL: 6

;; QUESTION SECTION:
;rsync.batten.eu.org.     IN AAAA

;; ANSWER SECTION:
rsync.batten.eu.org. 86400 IN AAAA 2001:630:c2:3263:8:20ff:fee9:4d41
```

AA = Authoritative Answer

```
ians-macbook-air:clocks igb$ dig aaaa rsync.batten.eu.org

; <<>> DiG 9.8.3-P1 <<>> aaaa rsync.batten.eu.org
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 232
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 0

;; QUESTION SECTION:
;rsync.batten.eu.org.     IN AAAA

;; ANSWER SECTION:
rsync.batten.eu.org. 86245 IN AAAA 2001:630:c2:3263:8:20ff:fee9:4d41
```

!AA

Full Initial Response

```
igb@offsite10:~$ dig aaaa mailstore.batten.eu.org

; <<>> DiG 9.16.33-Debian <<>> aaaa mailstore.batten.eu.org
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 5807
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 2, AUTHORITY: 4, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 1232
; COOKIE: 0e8206b2abb9ce800100000636373dabb74986906e5927f (good)
;; QUESTION SECTION:
;mailstore.batten.eu.org. IN AAAA

;; ANSWER SECTION:
mailstore.batten.eu.org. 300 IN CNAME offsite10.batten.eu.org.
offsite10.batten.eu.org. 86400  IN AAAA 2001:19f0:6c01:298f:5400:2ff:fe0f:c00a

;; AUTHORITY SECTION:
batten.eu.org.    86400 IN NS pns33.cloudns.net.
batten.eu.org.    86400 IN NS pns31.cloudns.net.
batten.eu.org.    86400 IN NS pns32.cloudns.net.
batten.eu.org.    86400 IN NS pns34.cloudns.net.

;; Query time: 0 msec
;; SERVER: ::1#53(::1)
;; WHEN: Thu Nov 03 07:55:06 GMT 2022
;; MSG SIZE  rcvd: 223

igb@offsite10:~$
```

“Here’s where I got it from, these are the sources of truth”

EDNS(0)

- There is an extension mechanism for DNS, called EDNS(0), set out in RFC6891. It is complex and messy, gives rise to some amplification attacks, and may not play nicely with some firewalls.
- It does have support for “cookies” which are large replacements for the transaction ID

Additional Information, from the cache

```
igb@offsite10:~$ dig a pns31.cloudns.net.
[…]
pns31.cloudns.net.   172800   IN AAAA2a06:fb00:1::1:66
[…]
igb@offsite10:~$ dig a pns31.cloudns.net.
[…]
pns31.cloudns.net.   172800   IN A  185.136.96.66
[…]
igb@offsite10:~$ dig aaaa mailstore.batten.eu.org

[…]
;mailstore.batten.eu.org.  IN AAAA

;; ANSWER SECTION:
mailstore.batten.eu.org. 300 IN CNAME offsite10.batten.eu.org.
offsite10.batten.eu.org. 86400  IN AAAA 2001:19f0:6c01:298f:5400:2ff:fe0f:c00a

;; AUTHORITY SECTION:
batten.eu.org.    86400 IN NS pns34.cloudns.net.
batten.eu.org.    86400 IN NS pns32.cloudns.net.
batten.eu.org.    86400 IN NS pns31.cloudns.net.
batten.eu.org.    86400 IN NS pns33.cloudns.net.

;; ADDITIONAL SECTION:
pns31.cloudns.net.   172794   IN AAAA2a06:fb00:1::1:66
pns31.cloudns.net.   172797   IN A  185.136.96.66

;; Query time: 0 msec
;; SERVER: ::1#53(::1)
;; WHEN: Thu Nov 03 07:56:09 GMT 2022
;; MSG SIZE  rcvd: 267
```

Here’s some handy stuff I happen to know, but I didn’t put any effort into finding it

Cached Response

```
igb@offsite10:~$ dig @1.1.1.1 offsite10.batten.eu.org

; <<>> DiG 9.16.33-Debian <<>> @1.1.1.1 offsite10.batten.eu.org
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 16099
;; flags: qr rd ra ad; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 1232
;; QUESTION SECTION:
;offsite10.batten.eu.org. IN A

;; ANSWER SECTION:
offsite10.batten.eu.org. 86395 IN A   209.250.236.89

;; Query time: 7 msec
;; SERVER: 1.1.1.1#53(1.1.1.1)
;; WHEN: Thu Nov 03 08:00:03 GMT 2022
;; MSG SIZE  rcvd: 68

igb@offsite10:~$
```

Command Line Example

```
ians-macbook-air:~ igb$ dig +norecurse @a.root-servers.net. ns uk | awk '$4=="NS"' | head -1
uk.		172800	IN	NS	nsd.nic.uk.
ians-macbook-air:~ igb$ dig +norecurse @nsd.nic.uk. ns ac.uk | awk '$4=="NS"' | head -1
ac.uk.		172800	IN	NS	ns1.surfnet.nl.
ians-macbook-air:~ igb$ dig +norecurse @ns1.surfnet.nl. ns bham.ac.uk | awk '$4=="NS"' |
head -1
bham.ac.uk.	86400	IN	NS	dns0.bham.ac.uk.
ians-macbook-air:~ igb$ dig +norecurse @dns0.bham.ac.uk. www.bham.ac.uk
```

You can experiment with this: sometimes the answer is returned as an answer, sometimes as authority records, depending the precise configuration of the server for the zone you are querying.

Command Line Example

```
ians-macbook-air:~ igb$ dig +norecurse @dns0.bham.ac.uk. www.bham.ac.uk

; <<>> DiG 9.8.3-P1 <<>> +norecurse @dns0.bham.ac.uk. www.bham.ac.uk
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 24883
;; flags: qr aa ra; QUERY: 1, ANSWER: 2, AUTHORITY: 2, ADDITIONAL: 2

;; QUESTION SECTION:
;www.bham.ac.uk.         IN A

;; ANSWER SECTION:
www.bham.ac.uk.    172800 IN CNAME corp501.bham.ac.uk.
corp501.bham.ac.uk.   172800 IN A  147.188.125.39

;; AUTHORITY SECTION:
bham.ac.uk.     172800 IN NS dns1.bham.ac.uk.
bham.ac.uk.     172800 IN NS dns0.bham.ac.uk.

;; ADDITIONAL SECTION:
dns0.bham.ac.uk.   172800 IN A  147.188.128.2
dns1.bham.ac.uk.   172800 IN A  147.188.128.102
```

Delegations

```
$ dig ns ac.uk | awk '$4=="NS"' | head -1
ac.uk.      28324 IN  NS  ws-fra1.win-ip.dfn.de.
```

```
$ dig @ws-fra1.win-ip.dfn.de. NS bham.ac.uk

; <<>> DiG 9.8.3-P1 <<>> @ws-fra1.win-ip.dfn.de. NS bham.ac.uk
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 11490
;; flags: qr rd; QUERY: 1, ANSWER: 0, AUTHORITY: 3, ADDITIONAL: 3
;; WARNING: recursion requested but not available

;; QUESTION SECTION:
;bham.ac.uk.         IN NS

;; AUTHORITY SECTION:
bham.ac.uk.    86400 IN NS dns0.bham.ac.uk.
bham.ac.uk.    86400 IN NS dncc01.bham.ac.uk.
bham.ac.uk.    86400 IN NS dns1.bham.ac.uk.

;; ADDITIONAL SECTION:
dns0.bham.ac.uk.  86400 IN A  147.188.128.2
dns1.bham.ac.uk.  86400 IN A  147.188.128.102
dncc01.bham.ac.uk.   86400 IN A  147.188.128.144

;; Query time: 23 msec
;; SERVER: 193.174.75.178#53(193.174.75.178)
;; WHEN: Wed Oct 26 23:29:32 2016
;; MSG SIZE  rcvd: 135
```

Authority Record

```
cs.bham.ac.uk.    86400 IN SOA dns0.cs.bham.ac.uk. hostmaster.cs.bham.ac.uk. (
          2015022000 ; serial
          10800      ; refresh (3 hours)
          3600       ; retry (1 hour)
          604800     ; expire (1 week)
          86400      ; minimum (1 day)
          )
```

Serial, Refresh, Retry and Expire for benefit of secondaries

Minimum now redefined for negative caching

NS record

Should match in bham.ac.uk and cs.bham.ac.uk zones

```
cs.bham.ac.uk.    86400 IN  NS dns1.cs.bham.ac.uk.
cs.bham.ac.uk.    86400 IN  NS dns0.cs.bham.ac.uk.
cs.bham.ac.uk.    86400 IN  NS dns0.bham.ac.uk.
cs.bham.ac.uk.    86400 IN  NS ns0.susx.ac.uk.
cs.bham.ac.uk.    86400 IN  NS ext-proxy.ftel.co.uk.
```

Additional Information

Note: incomplete

```
dns0.cs.bham.ac.uk. 86400   IN A 147.188.192.4
dns1.cs.bham.ac.uk. 86400   IN A 147.188.192.8
ext-proxy.ftel.co.uk. 86400   INA  192.65.220.99
ext-proxy.ftel.co.uk. 86400   INA  192.65.220.98
```

Old Attack

- Old nameserver software blindly accepted additional information and cached it
- Allows you to supply an IP number you control as “google.com 604800 IN A 1.2.3.4”; anyone who visits your nameserver has a chance of caching a bad nameserver for Google (or a bank, or whatever).
- Now stopped with “out of balliwick” controls: you only accept additional information that the server can reasonably be assumed to be authoritative for

Primary / Secondary

- DNS protocol has support for replicating zones between primary and secondary
  - Primary does not have to be visible, “hidden primary” is a popular pattern
  - Multiple nameservers can be updated by other means (SQL replication, rsync, people carrying USB sticks)

Pro Tip

- A common pattern on small networks is for the authoritative server to also be the recursive / caching server for local clients
- **DO NOT DO THIS.**

Delegation

- Suppose we have a nameserver for bham.ac.uk. How do we create the domain cs.bham.ac.uk on another nameserver (or at least another zone file, possibly with different access rules)?

Delegation: Just NS records

In bham.ac.ak

```
cs.bham.ac.uk.    28800  IN  NS  dns0.cs.bham.ac.uk.
cs.bham.ac.uk.    28800  IN  NS  dns1.cs.bham.ac.uk.
cs.bham.ac.uk.    28800  IN  NS  ns0.susx.ac.uk.
```

Glue Records

- How do you locate the A record for “dom.ain 86400 IN NS ns1.dom.ain”?
- The zonefile dom.ain in that case has an A record for ns1.dom.ain as well as an NS record for dom.ain.
- This is called a **Glue Record**

```
;; ADDITIONAL SECTION:
dns0.cs.bham.ac.uk.  28800 IN A 147.188.192.4
dns1.cs.bham.ac.uk.  28800 IN A 147.188.192.8
```

Mail Exchangers

```
;; ANSWER SECTION:
batten.eu.org.    86400  IN MX 4 bham-mx2.bham.ac.uk.
batten.eu.org.    86400   IN MX 4 bham-mx3.bham.ac.uk.
batten.eu.org.    86400 IN  MX 2 mail.batten.eu.org.
batten.eu.org.    86400   IN MX 4 bham-mx1.bham.ac.uk.

;; ADDITIONAL SECTION:
bham-mx2.bham.ac.uk. 86350 IN A  147.188.128.219
bham-mx3.bham.ac.uk. 86350 IN A  147.188.128.221
mail.batten.eu.org.  300   IN A  147.188.192.250
mail.batten.eu.org.  300   IN AAAA 2001:630:c2:3263:8:20ff:fed7:92ef
bham-mx1.bham.ac.uk. 86350 IN A  147.188.128.129
```

Zone File Maintenance

- You can edit zone-files manually, but it is very prone to error
- Most sites generate the zone files from some other source of information, usually a database or some XML (classic “greybeard” shell scripts, which scare everyone once the author leaves)
- Also dynamic DNS

Reverse Mapping

- IP number is transformed into reverse order and looked up in special in-addr.arpa or .ip6.arpa domain:

```
250.192.188.147.in-addr.arpa. 86400 IN PTR   offsite.batten.eu.org.

1.0.0.0.9.5.3.5.6.2.9.5.1.4.1.3.e.0.9.a.f.9.2.1.0.b.8.0.1.0.0.2.ip6.arpa. 86400  IN PTR pi-one.batten.eu.org.
```

Why the strange formats?

- in-addr.arpa format allows delegation on 8-bit boundaries.
- Makes delegation of /28s (say) difficult
- Various messy solutions: look them up.
- Lesson learnt, so ipv6 reverse mapping allows delegation on 4-bit boundaries in hierarchy.

Reverse Attacks

- Suppose I control 2.3.4.in-addr.arpa
- I can create RR “1.2.3.4.in-addr.arpa PTR something.bham.ac.uk” and try to pose as part of Birmingham network for purposes of libraries, Apple discounts, etc.
- Check is to look up something.bham.ac.uk and see if it matches: only bham.ac.uk admin can install required “something.bham.ac.uk A 1.2.3.4” record

DNS Security

- DNS Sec exists to sign zones, providing evidence that packets haven’t been tampered with
- Topic for network security lectures
  - Complex
  - Didn’t scale without major modifications
  - Very low adoption after ~20 years
  - Doesn’t solve common use-cases

Packets

Transaction ID	Flags	12 bytes
Question count	Answer RR count	
Authority RR count	Additional RR count	
Question entries (variable length)		Variable length
Answer RRs (variable length)		
Authority RRs (variable length)		
Additional RRs (variable length)		

16 bit transaction ID: serious problem
for security discovered in 2008 (“Kaminsky Attack”)

Kaminsky Attack

- DNS queries have a 16 bit qid, hopefully random (or not, in which case you are doomed)
- Attack is to flood recursive server with fake responses
- With high enough rates, you will sometimes force a collision for a common name, and can prime the DNS cache with your chosen records
- Less valuable in 2023 for google.com because of https, but still serious for targetted attacks against enterprises that are less careful
- Pro tip: never run an internal network with self-signed certificates

Kaminsky Attack

- Very old servers: all queries come **from** port 53
- Slightly less old servers: all queries come from one random port
- Modern servers: queries come from a randomly chosen port amongst a suite of 100 or more randomly chosen ports, effectively extending the qid by the bits in the port number: attacker now needs to guess port as well as qid.
  - (helped by nfd>20, often 1024 today, plus threading)
- This protection often undone by NAT: **DO NOT RUN A DNS RESOLVER BEHIND A NAT POINT** unless you really know what you are doing.
- Or…

DNS over HTTPS

- DNS as a Webapp, leveraging https security: RFC8484
- Terrifies ISPs, law enforcement, enterprises: see draft RFC https://datatracker.ietf.org/doc/draft-reid-doh-operator/
- Breaks quite a few censorship / parental control applications
- This year’s hot topic (both amongst operators and amongst law enforcement / governance people)

DoH

- Looks like a standard https request to port 443 on some server, so trying to block it is whack-a-mole.
- 1.1.1.1, 8.8.8.8, 9.9.9.9 are all offering it, but anyone can run a DoH gateway themselves
- Can either be a server?query=foo.dom.ain URL, or POST the raw bytes of a DNS packet (which is quite as mad as it sounds)

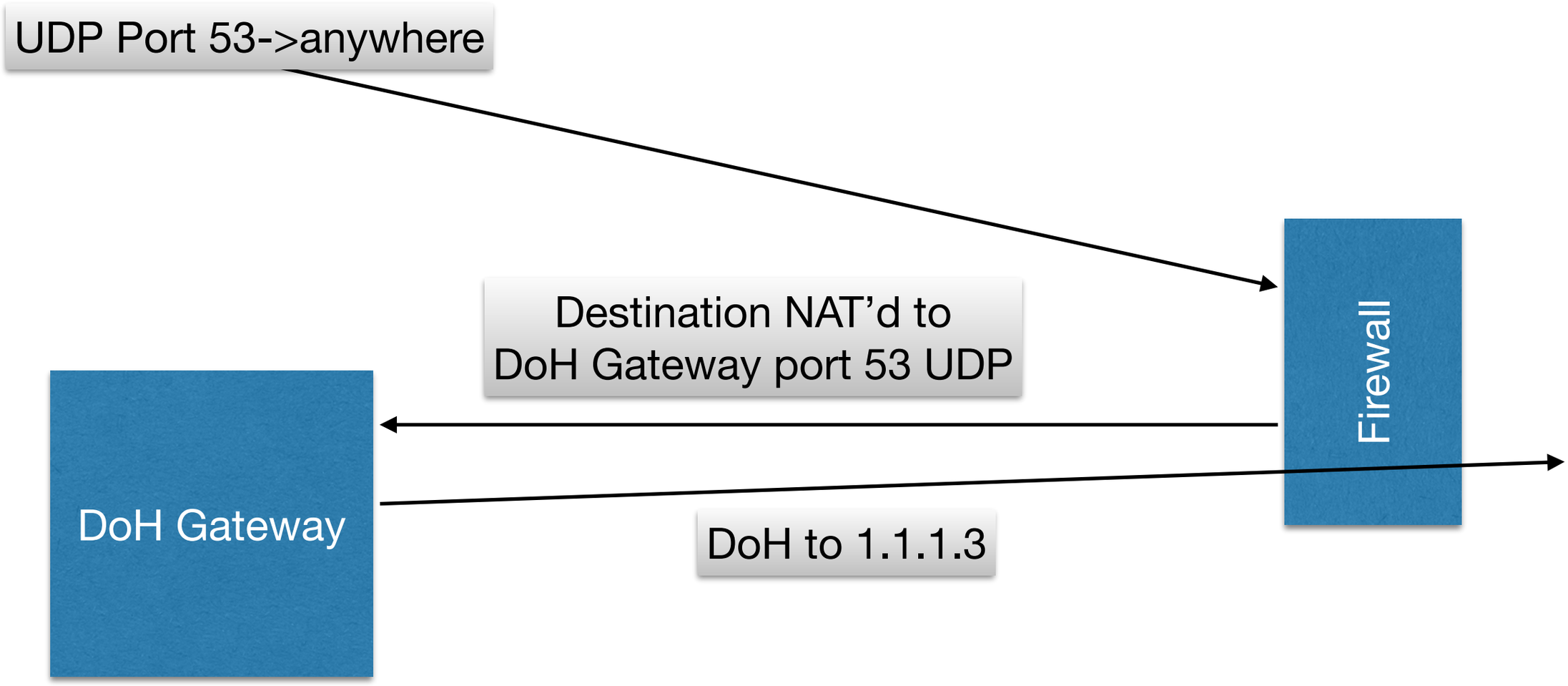
Problems

- ISP cannot gauge traffic
- Network owners cannot apply policy to users or, at least, it is a lot harder
  - A lot of “family” products like OpenDNS are just filtered DNS
  - Bypasses Response Policy Zones (look them up: https://dnsrpz.info/)
- Interception loses another insight (yeah, but sometimes there are actual bad people)

Solutions

- There aren’t any
- Chrome were persuaded to delay rolling out a DoH client built in to the browser in some territories (is this still delayed in UK?) and there are handles to allow management domains to turn it off
- Nonetheless, determined users can use it.

My House



Compliant clients use the gateway, random clients get NAT'd: this is a good usecase!

66582924