DNS

Next: back to IP addresses (again!)

DNS

Next: back to IP addresses (again!)

IP addresses (v4 or v6) are bunches of bits: but we are used to addressing machines by names such as `linux.bath.ac.uk`

DNS

Next: back to IP addresses (again!)

IP addresses (v4 or v6) are bunches of bits: but we are used to addressing machines by names such as `linux.bath.ac.uk`

The Internet would probably be a lot less easy to use if we had to refer to `212.58.233.253` rather than `www.bbc.co.uk`

DNS

Next: back to IP addresses (again!)

IP addresses (v4 or v6) are bunches of bits: but we are used to addressing machines by names such as `linux.bath.ac.uk`

The Internet would probably be a lot less easy to use if we had to refer to `212.58.233.253` rather than `www.bbc.co.uk`

The IP addresses are essential as they are hardware independent and have structure that aids routing

DNS

Next: back to IP addresses (again!)

IP addresses (v4 or v6) are bunches of bits: but we are used to addressing machines by names such as `linux.bath.ac.uk`

The Internet would probably be a lot less easy to use if we had to refer to `212.58.233.253` rather than `www.bbc.co.uk`

The IP addresses are essential as they are hardware independent and have structure that aids routing

But the names are what makes things like the Web usable

DNS

Next: back to IP addresses (again!)

IP addresses (v4 or v6) are bunches of bits: but we are used to addressing machines by names such as `linux.bath.ac.uk`

The Internet would probably be a lot less easy to use if we had to refer to `212.58.233.253` rather than `www.bbc.co.uk`

The IP addresses are essential as they are hardware independent and have structure that aids routing

But the names are what makes things like the Web usable

Note we now have *three* kinds of addresses: physical, network and human

DNS

So we have a repeat of the hardware-software address gap that ARP is for: there must also be a mechanism for turning `linux.bath.ac.uk` into `138.38.3.40`

DNS

So we have a repeat of the hardware-software address gap that ARP is for: there must also be a mechanism for turning `linux.bath.ac.uk` into `138.38.3.40`

In the early Internet every host had a file containing the names and addresses of all machines on the Internet

DNS

So we have a repeat of the hardware-software address gap that ARP is for: there must also be a mechanism for turning `linux.bath.ac.uk` into `138.38.3.40`

In the early Internet every host had a file containing the names and addresses of all machines on the Internet

A simple look into this table sufficed

DNS

So we have a repeat of the hardware-software address gap that ARP is for: there must also be a mechanism for turning `linux.bath.ac.uk` into `138.38.3.40`

In the early Internet every host had a file containing the names and addresses of all machines on the Internet

A simple look into this table sufficed

We can't do that now, though!

DNS

So we have a repeat of the hardware-software address gap that ARP is for: there must also be a mechanism for turning `linux.bath.ac.uk` into `138.38.3.40`

In the early Internet every host had a file containing the names and addresses of all machines on the Internet

A simple look into this table sufficed

We can't do that now, though!

Exercise The file lives on as `/etc/hosts` under Unix. Look at this file

DNS

An ARP-style discovery broadcast would have to go to the entire Internet, so this approach is also infeasible

DNS

An ARP-style discovery broadcast would have to go to the entire Internet, so this approach is also infeasible

So there is another protocol, the *domain name system* (DNS) to do this

DNS

An ARP-style discovery broadcast would have to go to the entire Internet, so this approach is also infeasible

So there is another protocol, the *domain name system* (DNS) to do this

There is no sensible, secure and economic way to have a single database that contains all the names and addresses on the Internet, so this information has to be distributed amongst a large number of machines called DNS servers

DNS

An ARP-style discovery broadcast would have to go to the entire Internet, so this approach is also infeasible

So there is another protocol, the *domain name system* (DNS) to do this

There is no sensible, secure and economic way to have a single database that contains all the names and addresses on the Internet, so this information has to be distributed amongst a large number of machines called DNS servers

That is, machines running a DNS server program

Aside

In the usual way, when talking about these things, we blur the distinction between the service and the host it lives on. The host might well be running other services, such as DCHP, as well

Aside

In the usual way, when talking about these things, we blur the distinction between the service and the host it lives on. The host might well be running other services, such as DCHP, as well

Exercise Find out the services your home Access Point runs

DNS

Names are hierarchical: `linux.bath.ac.uk` is a name of a machine in the *domain* `bath.ac.uk`

DNS

Names are hierarchical: `linux.bath.ac.uk` is a name of a machine in the *domain* `bath.ac.uk`

`bath.ac.uk` is a network in the domain `ac.uk`, and so on

DNS

Names are hierarchical: `linux.bath.ac.uk` is a name of a machine in the *domain* `bath.ac.uk`

`bath.ac.uk` is a network in the domain `ac.uk`, and so on

`ac.uk` is the name for the JANET network, and is in the domain `uk`

DNS

Names are hierarchical: `linux.bath.ac.uk` is a name of a machine in the *domain* `bath.ac.uk`

`bath.ac.uk` is a network in the domain `ac.uk`, and so on

`ac.uk` is the name for the JANET network, and is in the domain `uk`

`uk` is in the domain `.` (*dot* or *root*)

DNS

Names are hierarchical: `linux.bath.ac.uk` is a name of a machine in the *domain* `bath.ac.uk`

`bath.ac.uk` is a network in the domain `ac.uk`, and so on

`ac.uk` is the name for the JANET network, and is in the domain `uk`

`uk` is in the domain `.` (*dot* or *root*)

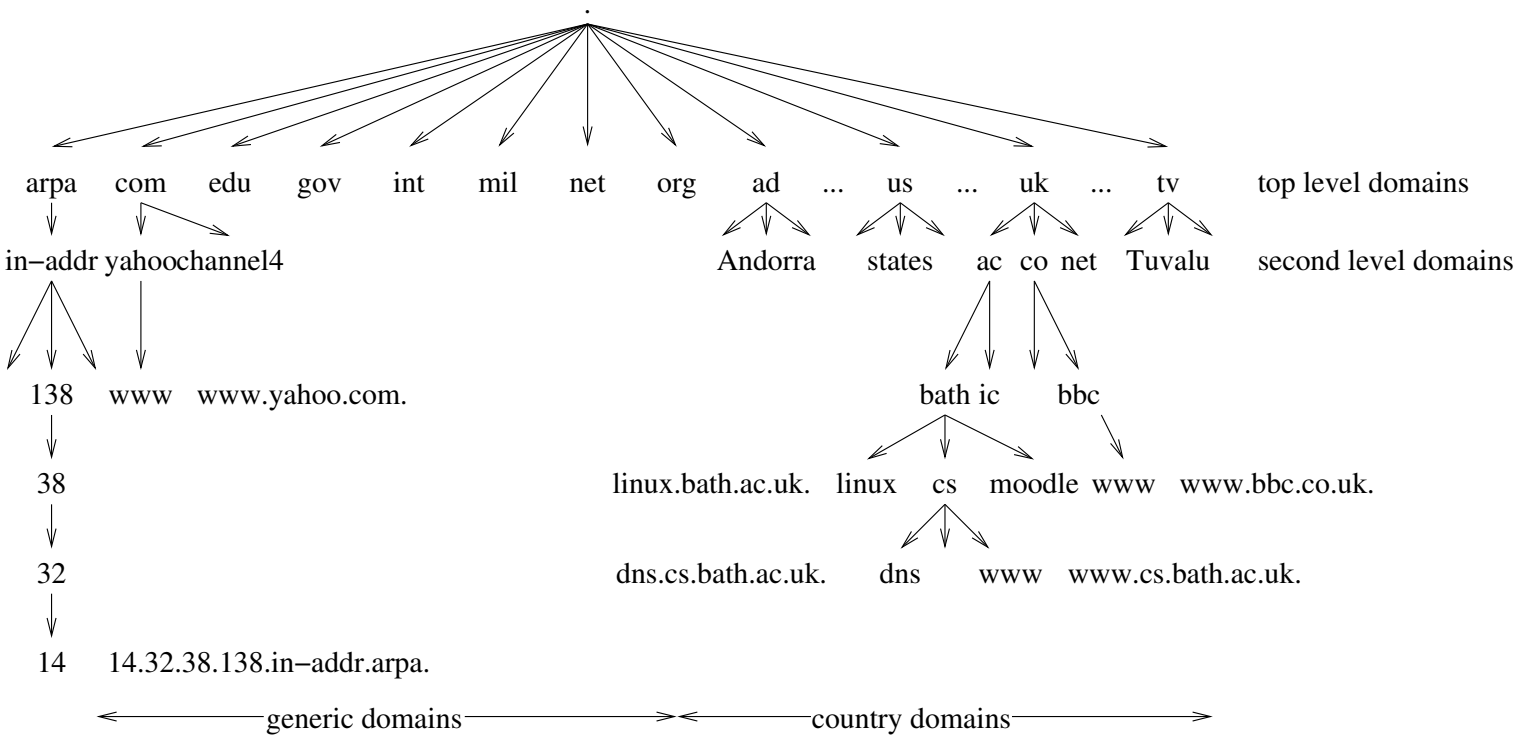
Each node in this tree is called a *label*

DNS

`www.llanfairpwllgwyngyllgogerychwyrndrobwllllantysiliogogogoch.com`
is a valid name

Labels may be up to 63 bytes long

DNS



The DNS hierarchy

DNS

It is this tree we use to distribute the database

DNS

It is this tree we use to distribute the database

The root of the tree is called . (dot) and this label is currently managed by the *Internet Corporation for Assigned Names and Numbers* (ICANN)

DNS

It is this tree we use to distribute the database

The root of the tree is called `.` (dot) and this label is currently managed by the *Internet Corporation for Assigned Names and Numbers* (ICANN)

They manage the *top level domains* (TLDs), including `com` and `uk`

DNS

It is this tree we use to distribute the database

The root of the tree is called `.` (dot) and this label is currently managed by the *Internet Corporation for Assigned Names and Numbers* (ICANN)

They manage the *top level domains* (TLDs), including `com` and `uk`

“Manage” means they keep the lookup tables for that level and they say who can get labels in the next level

DNS

It is this tree we use to distribute the database

The root of the tree is called `.` (dot) and this label is currently managed by the *Internet Corporation for Assigned Names and Numbers* (ICANN)

They manage the *top level domains* (TLDs), including `com` and `uk`

“Manage” means they keep the lookup tables for that level and they say who can get labels in the next level

They get to charge money for labels in the next level

DNS

It is this tree we use to distribute the database

The root of the tree is called `.` (dot) and this label is currently managed by the *Internet Corporation for Assigned Names and Numbers* (ICANN)

They manage the *top level domains* (TLDs), including `com` and `uk`

“Manage” means they keep the lookup tables for that level and they say who can get labels in the next level

They get to charge money for labels in the next level

But the important bit is that they *delegate* management of lower labels to other organisations

DNS

Labels under `uk` are managed by the UK *Network Information Centre* (NIC), currently run by a UK company called Nominet

DNS

Labels under `uk` are managed by the UK *Network Information Centre* (NIC), currently run by a UK company called Nominet

Note that ICANN gets to say that Nominet is in charge of `uk`

DNS

Labels under `uk` are managed by the UK *Network Information Centre* (NIC), currently run by a UK company called Nominet

Note that ICANN gets to say that Nominet is in charge of `uk`

Nominet delegates labels under `ac.uk` to the *Joint Information Systems Committee* (Jisc) (previously *United Kingdom Education and Research Networking Association* (UKERNA))

DNS

Labels under `uk` are managed by the UK *Network Information Centre* (NIC), currently run by a UK company called Nominet

Note that ICANN gets to say that Nominet is in charge of `uk`

Nominet delegates labels under `ac.uk` to the *Joint Information Systems Committee* (Jisc) (previously *United Kingdom Education and Research Networking Association* (UKERNA))

Labels under `bath.ac.uk` are managed by DDAT for the University of Bath

DNS

Labels under `uk` are managed by the UK *Network Information Centre* (NIC), currently run by a UK company called Nominet

Note that ICANN gets to say that Nominet is in charge of `uk`

Nominet delegates labels under `ac.uk` to the *Joint Information Systems Committee* (Jisc) (previously *United Kingdom Education and Research Networking Association* (UKERNA))

Labels under `bath.ac.uk` are managed by DDAT for the University of Bath

Labels under `cs.bath.ac.uk` are managed by the ~~Department of Computer Science~~ DDAT

DNS

At each level, the relevant organisation keeps a list of labels it has delegated and who is responsible for them

DNS

At each level, the relevant organisation keeps a list of labels it has delegated and who is responsible for them

At the lowest levels, for the leaves of the tree (the hosts), the relevant organisation keeps the name-to-IP-address mapping of the hosts

DNS

At each level, the relevant organisation keeps a list of labels it has delegated and who is responsible for them

At the lowest levels, for the leaves of the tree (the hosts), the relevant organisation keeps the name-to-IP-address mapping of the hosts

So, starting a dot, we can work our way down the tree to find the machine we want

DNS

All this is done by *name servers*

DNS

All this is done by *name servers*

These are just computers that run the DNS protocol

DNS

All this is done by *name servers*

These are just computers that run the DNS protocol

E.g., `ns1.bath.ac.uk` is (the name of) a name server for the University

DNS

All this is done by *name servers*

These are just computers that run the DNS protocol

E.g., `ns1.bath.ac.uk` is (the name of) a name server for the University

Also there is `ns2.bath.ac.uk`, for resilience and spreading load

DNS

All this is done by *name servers*

These are just computers that run the DNS protocol

E.g., `ns1.bath.ac.uk` is (the name of) a name server for the University

Also there is `ns2.bath.ac.uk`, for resilience and spreading load

They contain replicas of the same information

DNS

All this is done by *name servers*

These are just computers that run the DNS protocol

E.g., `ns1.bath.ac.uk` is (the name of) a name server for the University

Also there is `ns2.bath.ac.uk`, for resilience and spreading load

They contain replicas of the same information

A few years ago, there was a convention of having an off-site replica, too, e.g., Bath used `ns2.ja.net`

DNS

DNS servers do two things

DNS

DNS servers do two things

(a) Keep the table of name to IP address mappings for the domain

DNS

DNS servers do two things

(a) Keep the table of name to IP address mappings for the domain

(b) Help the local hosts when they need to do a lookup of non-local addresses

DNS

DNS servers do two things

(a) Keep the table of name to IP address mappings for the domain

(b) Help the local hosts when they need to do a lookup of non-local addresses

The former is “just” a lookup in a database

DNS

DNS servers do two things

(a) Keep the table of name to IP address mappings for the domain

(b) Help the local hosts when they need to do a lookup of non-local addresses

The former is “just” a lookup in a database

If a host on the Bath network needs to look up a local name, `linux.bath.ac.uk`, say, it sends a request to one of the local servers, e.g., `ns2.bath.ac.uk`

DNS

DNS servers do two things

(a) Keep the table of name to IP address mappings for the domain

(b) Help the local hosts when they need to do a lookup of non-local addresses

The former is “just” a lookup in a database

If a host on the Bath network needs to look up a local name, `linux.bath.ac.uk`, say, it sends a request to one of the local servers, e.g., `ns2.bath.ac.uk`

And the local server will look it up and return the answer

DNS

How does that host know which machine to ask? In particular, how does it know the local DNS server's IP address?

DNS

How does that host know which machine to ask? In particular, how does it know the local DNS server's IP address?

Every host has the IP addresses of one or more local nameservers, typically in a file (e.g., `/etc/resolv.conf`) that was set up by an administrator or was created by DHCP (or SLAAC)

DNS

How does that host know which machine to ask? In particular, how does it know the local DNS server's IP address?

Every host has the IP addresses of one or more local nameservers, typically in a file (e.g., `/etc/resolv.conf`) that was set up by an administrator or was created by DHCP (or SLAAC)

So it can send the DNS request to one of these servers

DNS

The second function is more interesting: we shall look at an example

DNS

The second function is more interesting: we shall look at an example

If a host in the University requires a name lookup of a non-local name, say `news.bbc.co.uk`, it sends a DNS request to the **local** DNS server, `ns1.bath.ac.uk`, say

DNS

In this example, the Bath server does not have responsibility for the name `news.bbc.co.uk`, but it will helpfully do a lookup for us. It will run down the DNS tree in a *recursive lookup*

DNS

In this example, the Bath server does not have responsibility for the name `news.bbc.co.uk`, but it will helpfully do a lookup for us. It will run down the DNS tree in a *recursive lookup*

The Bath server `ns1` sends a *start of authority* (SOA) request to a random *top level* server to find who is responsible for the `uk` label

DNS

At the top level, there are about 80 machines dotted about the world that serve the DNS root; they all have identical information

DNS

At the top level, there are about 80 machines dotted about the world that serve the DNS root; they all have identical information

They have names like `a.root-servers.net` (`198.41.0.4`), `b.root-servers.net` (`170.247.170.2`), and so on

DNS

At the top level, there are about 80 machines dotted about the world that serve the DNS root; they all have identical information

They have names like `a.root-servers.net` (`198.41.0.4`), `b.root-servers.net` (`170.247.170.2`), and so on

And the local nameserver has the IP addresses of these machines in a file: otherwise we would never get started!

DNS

At the top level, there are about 80 machines dotted about the world that serve the DNS root; they all have identical information

They have names like `a.root-servers.net` (`198.41.0.4`), `b.root-servers.net` (`170.247.170.2`), and so on

And the local nameserver has the IP addresses of these machines in a file: otherwise we would never get started!

The selected top level server responds to the Bath nameserver with something like `ns1.nic.uk`, together with its IP address

DNS

At the top level, there are about 80 machines dotted about the world that serve the DNS root; they all have identical information

They have names like `a.root-servers.net` (`198.41.0.4`), `b.root-servers.net` (`170.247.170.2`), and so on

And the local nameserver has the IP addresses of these machines in a file: otherwise we would never get started!

The selected top level server responds to the Bath nameserver with something like `ns1.nic.uk`, together with its IP address

This a machine that is responsible for `uk` labels

DNS

The Bath server now sends a SOA request to `ns1.nic.uk` for `co.uk`

DNS

The Bath server now sends a SOA request to `ns1.nic.uk` for `co.uk`

It gets a reply `ns1.nic.uk`. It just so happens this is responsible for both `uk` and `co.uk`

DNS

The Bath server now sends a SOA request to `ns1.nic.uk` for `co.uk`

It gets a reply `ns1.nic.uk`. It just so happens this is responsible for both `uk` and `co.uk`

So the server sends a SOA request to `ns1.nic.uk` for `bbc.co.uk`

DNS

The Bath server now sends a SOA request to `ns1.nic.uk` for `co.uk`

It gets a reply `ns1.nic.uk`. It just so happens this is responsible for both `uk` and `co.uk`

So the server sends a SOA request to `ns1.nic.uk` for `bbc.co.uk`

It gets `ns.bbc.co.uk`

DNS

It now knows the server reponsible for the address we want

DNS

It now knows the server reponsible for the address we want

The server sends an *address* (A) request for `news.bbc.co.uk` to `ns.bbc.co.uk`

DNS

It now knows the server reponsible for the address we want

The server sends an *address* (A) request for `news.bbc.co.uk` to `ns.bbc.co.uk`

It gets the IP address `194.130.56.40`

DNS

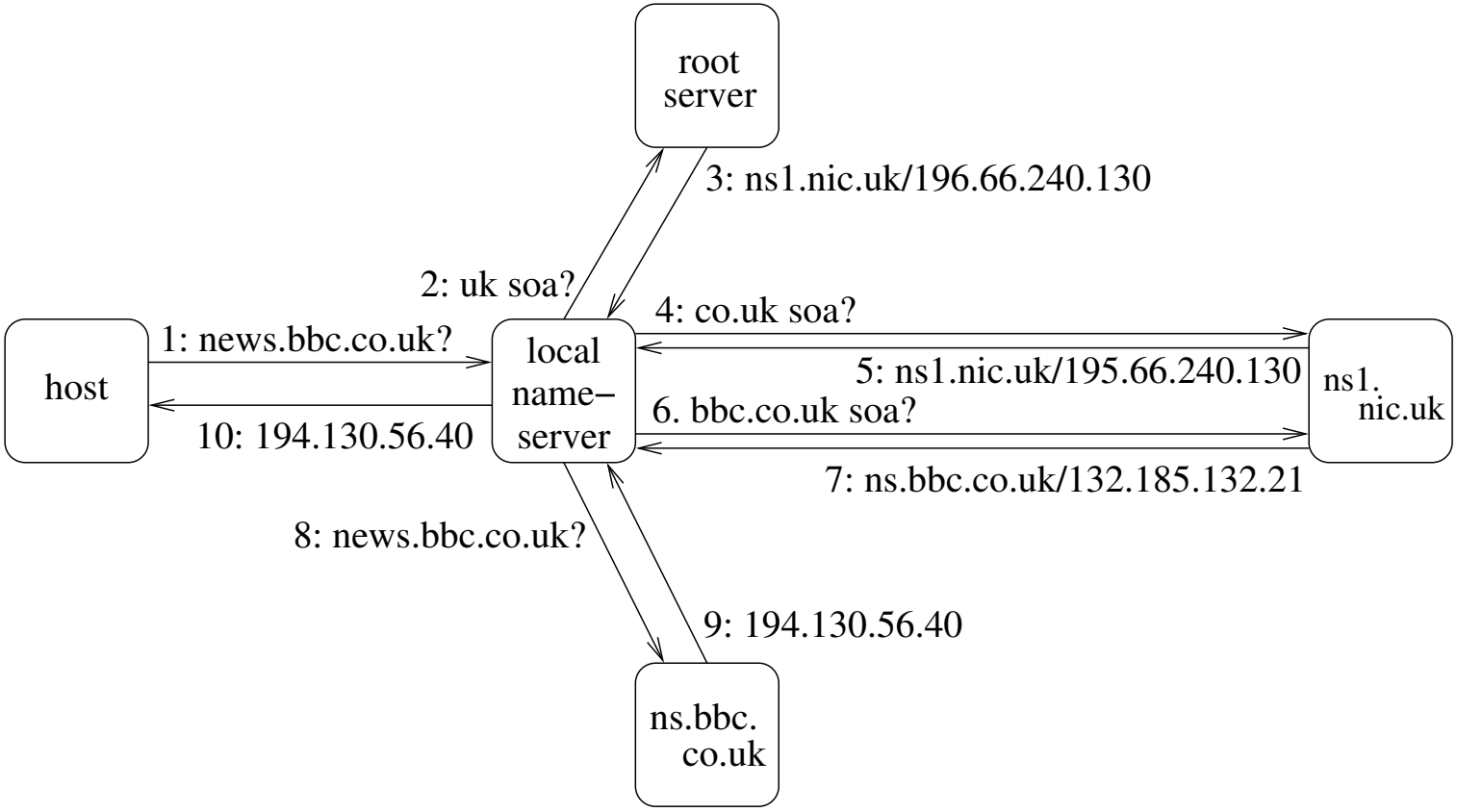
It now knows the server reponsible for the address we want

The server sends an *address* (A) request for `news.bbc.co.uk` to `ns.bbc.co.uk`

It gets the IP address `194.130.56.40`

Finally, the Bath server can relay this back to the original requesting host

DNS



DNS lookup

DNS

Of course, all these responses are cached by the local server so that it doesn't have to go through a complete lookup every time

DNS

Of course, all these responses are cached by the local server so that it doesn't have to go through a complete lookup every time

The next request for `news.bbc.co.uk` can be answered directly by the local nameserver

DNS

Of course, all these responses are cached by the local server so that it doesn't have to go through a complete lookup every time

The next request for `news.bbc.co.uk` can be answered directly by the local nameserver

Similarly, given a request for `www.bbc.co.uk`, it can go directly to `ns.bbc.co.uk`

DNS

As name mappings are not permanent, each reply has a *time to live* attached to it: this indicates how long the server should keep the information before asking again

DNS

As name mappings are not permanent, each reply has a *time to live* attached to it: this indicates how long the server should keep the information before asking again

Stable, long lived associations will have a long TTL

DNS

As name mappings are not permanent, each reply has a *time to live* attached to it: this indicates how long the server should keep the information before asking again

Stable, long lived associations will have a long TTL

Short lived associations will have a short TTL

DNS

The original asking host could do this lookup process itself but the benefit of using the organisation's name server is that it might well have done some or all of the steps already for some other request and cached them

DNS

The original asking host could do this lookup process itself but the benefit of using the organisation's name server is that it might well have done some or all of the steps already for some other request and cached them

This means a faster response and a decrease in network traffic

DNS

The original asking host could do this lookup process itself but the benefit of using the organisation's name server is that it might well have done some or all of the steps already for some other request and cached them

This means a faster response and a decrease in network traffic

And less work for the host

DNS

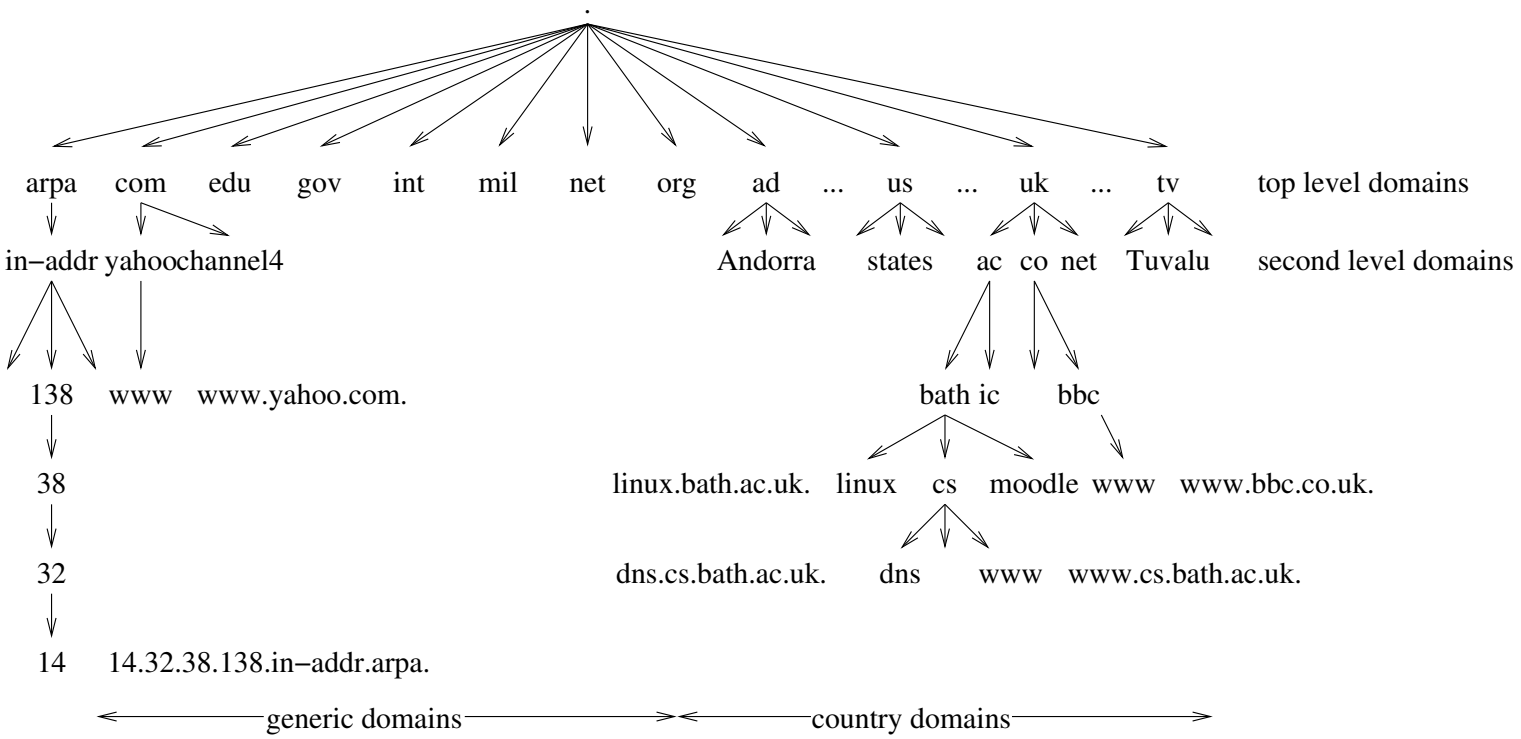
Sometimes we want to do the reverse lookup: given an IP address find a name

DNS

Sometimes we want to do the reverse lookup: given an IP address find a name

There is a part of the tree dedicated to this with TLD `arpa`

DNS



The DNS hierarchy

DNS

If we have the IP address `138.38.32.14` we can do a request for `14.32.38.138.in-addr.arpa`

DNS

If we have the IP address `138.38.32.14` we can do a request for `14.32.38.138.in-addr.arpa`

The same kind of recursive lookup as before will lead us to finding that Bath is responsible for `38.138.in-addr.arpa`

DNS

If we have the IP address `138.38.32.14` we can do a request for `14.32.38.138.in-addr.arpa`

The same kind of recursive lookup as before will lead us to finding that Bath is responsible for `38.138.in-addr.arpa`

And a couple more steps takes us to a *pointer* (PTR) request for `14.32.38.138.in-addr.arpa` from the Bath server

DNS

If we have the IP address `138.38.32.14` we can do a request for `14.32.38.138.in-addr.arpa`

The same kind of recursive lookup as before will lead us to finding that Bath is responsible for `38.138.in-addr.arpa`

And a couple more steps takes us to a *pointer* (PTR) request for `14.32.38.138.in-addr.arpa` from the Bath server

We get `clan.bath.ac.uk`

DNS

DNS does

DNS

DNS does

- A address: name to IP address

DNS

DNS does

- A address: name to IP address
- PTR pointer: IP address to name (IPv6 uses the `ip6.arpa` branch)

DNS

DNS does

- A address: name to IP address
- PTR pointer: IP address to name (IPv6 uses the `ip6.arpa` branch)
- AAAA address: name to IPv6 address

DNS

DNS does

- A address: name to IP address
- PTR pointer: IP address to name (IPv6 uses the `ip6.arpa` branch)
- AAAA address: name to IPv6 address
- SOA start of authority: name to responsible name server

DNS

DNS does

- A address: name to IP address
- PTR pointer: IP address to name (IPv6 uses the `ip6.arpa` branch)
- AAAA address: name to IPv6 address
- SOA start of authority: name to responsible name server
- MX mail server: name to a mail server for that domain. E.g., `bath.ac.uk` used to have mail server `138.38.32.14`

DNS

DNS does

- A address: name to IP address
- PTR pointer: IP address to name (IPv6 uses the `ip6.arpa` branch)
- AAAA address: name to IPv6 address
- SOA start of authority: name to responsible name server
- MX mail server: name to a mail server for that domain. E.g., `bath.ac.uk` used to have mail server `138.38.32.14`

And many more: about 50 in total, though few are used frequently

DNS

DNS is remarkably successful and scales easily to the billions of hosts on the Internet

DNS

DNS is remarkably successful and scales easily to the billions of hosts on the Internet

When a new organisation plugs into the Internet, it brings along its own DNS servers, thus increasing the scale of DNS

DNS

DNS is remarkably successful and scales easily to the billions of hosts on the Internet

When a new organisation plugs into the Internet, it brings along its own DNS servers, thus increasing the scale of DNS

Lookup usually takes a few milliseconds: caching of all levels in the local DNS server is a big help here

DNS

DNS is remarkably successful and scales easily to the billions of hosts on the Internet

When a new organisation plugs into the Internet, it brings along its own DNS servers, thus increasing the scale of DNS

Lookup usually takes a few milliseconds: caching of all levels in the local DNS server is a big help here

DNS is actually a many-many relationship of names and addresses

DNS

DNS is remarkably successful and scales easily to the billions of hosts on the Internet

When a new organisation plugs into the Internet, it brings along its own DNS servers, thus increasing the scale of DNS

Lookup usually takes a few milliseconds: caching of all levels in the local DNS server is a big help here

DNS is actually a many-many relationship of names and addresses

- One address can have several names

DNS

Both `news.bbc.co.uk` and `newswww.bbc.net.uk` resolve to `212.58.226.33`

```
news.bbc.co.uk.          1619    IN  CNAME   newswww.bbc.net.uk.
newswww.bbc.net.uk.      77      IN  A       212.58.226.33
```

DNS

Both `news.bbc.co.uk` and `newswww.bbc.net.uk` resolve to `212.58.226.33`

```
news.bbc.co.uk.          1619    IN  CNAME   newswww.bbc.net.uk.
newswww.bbc.net.uk.      77      IN  A       212.58.226.33
```

`news.bbc.co.uk` is an *alias* for the real *canonical name* `newswww.bbc.net.uk`

DNS

Both `news.bbc.co.uk` and `newswww.bbc.net.uk` resolve to `212.58.226.33`

```
news.bbc.co.uk.          1619    IN  CNAME   newswww.bbc.net.uk.
newswww.bbc.net.uk.      77      IN  A       212.58.226.33
```

`news.bbc.co.uk` is an *alias* for the real *canonical name* `newswww.bbc.net.uk`

This allows us tricks, like moving the Web server to different hosts, even different ISPs or different countries, but keeping its public name the same

DNS

- One name can have several IP addresses associated. This allows *load sharing*, e.g., a Web server can actually be several machines spread about anywhere in the world

DNS

```
www.microsoft.com.        68      IN  CNAME   toggle.www.ms.akadns.net.
toggle.www.ms.akadns.net. 285     IN  CNAME   g.www.ms.akadns.net.
g.www.ms.akadns.net.      285     IN  CNAME   lb1.www.ms.akadns.net.
lb1.www.ms.akadns.net.    285     IN  A       207.46.19.190
lb1.www.ms.akadns.net.    285     IN  A       207.46.19.254
lb1.www.ms.akadns.net.    285     IN  A       207.46.192.254
lb1.www.ms.akadns.net.    285     IN  A       207.46.193.254
```

DNS

```
www.microsoft.com.       68      IN  CNAME   toggle.www.ms.akadns.net.
toggle.www.ms.akadns.net. 285    IN  CNAME   g.www.ms.akadns.net.
g.www.ms.akadns.net.     285     IN  CNAME   lb1.www.ms.akadns.net.
lb1.www.ms.akadns.net.   285     IN  A       207.46.19.190
lb1.www.ms.akadns.net.   285     IN  A       207.46.19.254
lb1.www.ms.akadns.net.   285     IN  A       207.46.192.254
lb1.www.ms.akadns.net.   285     IN  A       207.46.193.254
```

`www.microsoft.com` is an alias: its canonical name is `toggle.www.ms.akadns.net`;

DNS

```
www.microsoft.com.       68      IN  CNAME   toggle.www.ms.akadns.net.
toggle.www.ms.akadns.net. 285    IN  CNAME   g.www.ms.akadns.net.
g.www.ms.akadns.net.     285     IN  CNAME   lb1.www.ms.akadns.net.
lb1.www.ms.akadns.net.   285     IN  A       207.46.19.190
lb1.www.ms.akadns.net.   285     IN  A       207.46.19.254
lb1.www.ms.akadns.net.   285     IN  A       207.46.192.254
lb1.www.ms.akadns.net.   285     IN  A       207.46.193.254
```

But that is an alias for `g.www.ms.akadns.net`;

DNS

```
www.microsoft.com.        68      IN  CNAME   toggle.www.ms.akadns.net.
toggle.www.ms.akadns.net. 285     IN  CNAME   g.www.ms.akadns.net.
g.www.ms.akadns.net.      285     IN  CNAME   lb1.www.ms.akadns.net.
lb1.www.ms.akadns.net.    285     IN  A       207.46.19.190
lb1.www.ms.akadns.net.    285     IN  A       207.46.19.254
lb1.www.ms.akadns.net.    285     IN  A       207.46.192.254
lb1.www.ms.akadns.net.    285     IN  A       207.46.193.254
```

And that is an alias for `lb1.www.ms.akadns.net`;

DNS

```
www.microsoft.com.        68      IN  CNAME   toggle.www.ms.akadns.net.
toggle.www.ms.akadns.net. 285     IN  CNAME   g.www.ms.akadns.net.
g.www.ms.akadns.net.      285     IN  CNAME   lb1.www.ms.akadns.net.
lb1.www.ms.akadns.net.    285     IN  A       207.46.19.190
lb1.www.ms.akadns.net.    285     IN  A       207.46.19.254
lb1.www.ms.akadns.net.    285     IN  A       207.46.192.254
lb1.www.ms.akadns.net.    285     IN  A       207.46.193.254
```

And that refers to four different addresses

DNS

And a DNS server can give out different lookups dependent on who is asking

DNS

And a DNS server can give out different lookups dependent on who is asking

A recent lookup for `www.microsoft.com` returned `104.78.177.250` from one location, and `2.18.85.172` from another

DNS

And a DNS server can give out different lookups dependent on who is asking

A recent lookup for `www.microsoft.com` returned `104.78.177.250` from one location, and `2.18.85.172` from another

This allows for load sharing; and also can be used for *geofencing*: giving different services to clients in different place, e.g., videos that are licensed only for certain regions

DNS

Exercise Redo these lookups to see how they currently turn out

Exercise Compare having DNS give out multiple IP addresses for a given name against giving out different addresses for different clients against using anycast addresses

DNS

In C programs, if we need to look up an address (v4 or v6), we can use a simple function call to `getaddrinfo()` that hides all this complexity from us

DNS

In C programs, if we need to look up an address (v4 or v6), we can use a simple function call to `getaddrinfo()` that hides all this complexity from us

Other languages will have similar APIs

DNS

In C programs, if we need to look up an address (v4 or v6), we can use a simple function call to `getaddrinfo()` that hides all this complexity from us

Other languages will have similar APIs

Exercise Old code using the obsolete v4-specific `gethostbyname()` is one of the sticking points in the transition to IPv6. Read about this

DNS

In C programs, if we need to look up an address (v4 or v6), we can use a simple function call to `getaddrinfo()` that hides all this complexity from us

Other languages will have similar APIs

Exercise Old code using the obsolete v4-specific `gethostbyname()` is one of the sticking points in the transition to IPv6. Read about this

Under Linux the `dig` tool can be used to do direct lookups

DNS

```
% dig news.bbc.co.uk
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 15281
;; flags: qr rd ra; QUERY: 1, ANSWER: 2, AUTHORITY: 2, ADDITIONAL: 2

;; QUESTION SECTION:
;news.bbc.co.uk.                        IN      A

;; ANSWER SECTION:
news.bbc.co.uk.         193     IN      CNAME   newswww.bbc.net.uk.
newswww.bbc.net.uk.     76      IN      A       212.58.226.73

;; AUTHORITY SECTION:
bbc.net.uk.             85129   IN      NS      ns0.thdo.bbc.co.uk.
bbc.net.uk.             85129   IN      NS      ns0.rbsov.bbc.co.uk.

;; ADDITIONAL SECTION:
ns0.thdo.bbc.co.uk.     9490    IN      A       212.58.224.20
ns0.rbsov.bbc.co.uk.    9490    IN      A       212.58.227.47
```

DNS

Quite often a DNS server will reply with more information than we requested, e.g., the lookup of the CNAME `newswww.bbc.net.uk` in the above

DNS

Quite often a DNS server will reply with more information than we requested, e.g., the lookup of the CNAME `newswww.bbc.net.uk` in the above

This means we don't have to do an additional query to get the actual IP address we were looking for

DNS

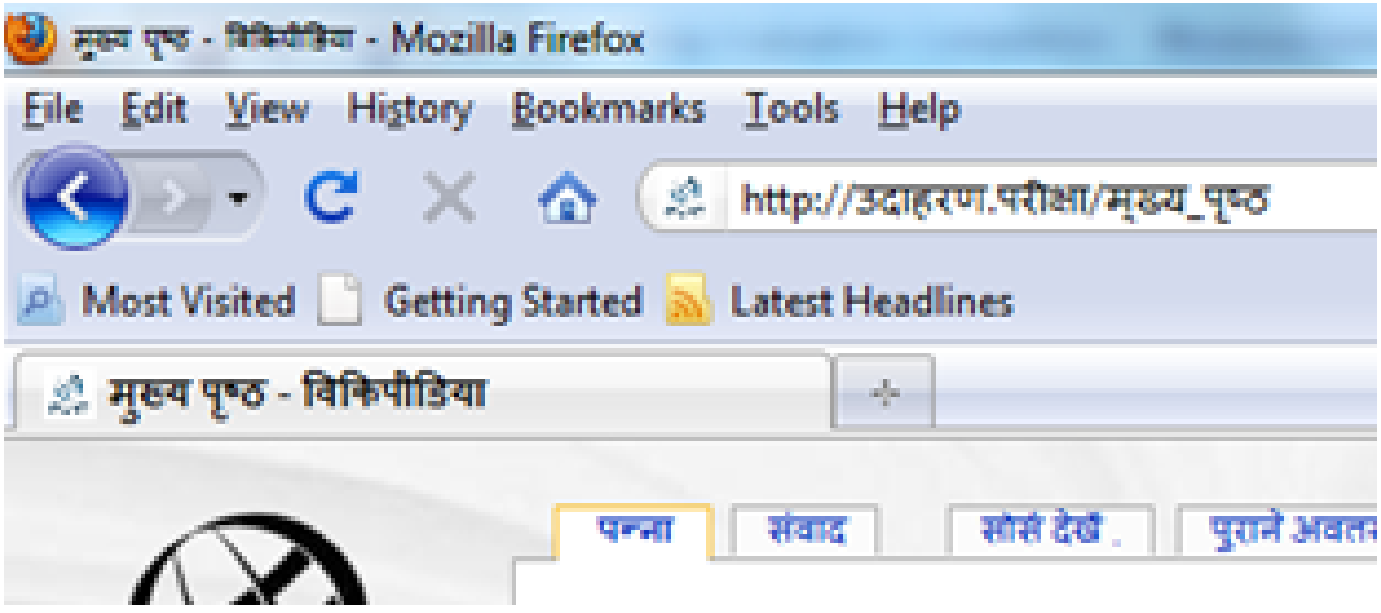
```
% dig -x 212.58.226.73
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 32413
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 4, ADDITIONAL: 3

;; QUESTION SECTION:
;73.226.58.212.in-addr.arpa.    IN      PTR

;; ANSWER SECTION:
73.226.58.212.in-addr.arpa. 50081 IN    PTR     newslb305.telhc.bbc.co.uk.

;; AUTHORITY SECTION:
226.58.212.in-addr.arpa. 50081  IN      NS      ns1.thdo.bbc.co.uk.
226.58.212.in-addr.arpa. 50081  IN      NS      ns1.thny.bbc.co.uk.
226.58.212.in-addr.arpa. 50081  IN      NS      ns1.bbc.co.uk.
226.58.212.in-addr.arpa. 50081  IN      NS      ns.bbc.co.uk.

;; ADDITIONAL SECTION:
ns1.bbc.co.uk.          311     IN      A       132.185.132.21
ns1.thny.bbc.co.uk.     32106   IN      A       212.58.227.48
ns1.thdo.bbc.co.uk.     33051   IN      A       212.58.224.21
```

DNS

```
% dig +trace www.google.com
.                       180695  IN      NS      E.ROOT-SERVERS.NET.
.                       180695  IN      NS      J.ROOT-SERVERS.NET.
.                       180695  IN      NS      I.ROOT-SERVERS.NET.
...
com.                    172800  IN      NS      A.GTLD-SERVERS.NET.
com.                    172800  IN      NS      B.GTLD-SERVERS.NET.
com.                    172800  IN      NS      C.GTLD-SERVERS.NET.
...
google.com.             172800  IN      NS      ns1.google.com.
google.com.             172800  IN      NS      ns2.google.com.
google.com.             172800  IN      NS      ns3.google.com.
...
www.google.com.         604800  IN      CNAME   www.l.google.com.
l.google.com.           86400   IN      NS      b.l.google.com.
l.google.com.           86400   IN      NS      d.l.google.com.
...
```

DNS

DNS labels can be in arbitrary character sets, not just Latin



A non-Latin DNS name

From `blog.icann.org`

DNS

DNS is a very successful protocol: fast and it's relatively easy for system administrators to manage the DNS servers

DNS

DNS is a very successful protocol: fast and it's relatively easy for system administrators to manage the DNS servers

DNS names are a big source of money, so often a source of contention over who should be in control of what

DNS

DNS is a very successful protocol: fast and it's relatively easy for system administrators to manage the DNS servers

DNS names are a big source of money, so often a source of contention over who should be in control of what

Exercise Read about the controversies behind the introduction of new top-level DNS labels like `me` and `search`

DNS

Exercise Try looking up

- AAAA for `bath.ac.uk`
- AAAA for `facebook.com`
- SOA for `bath.ac.uk`
- A for `moodle.bath.ac.uk`
- And so on

DNS

Exercise And

- MX for `bath.ac.uk`

What is the Uni's mail server physical location?

What are the privacy/security/legal aspects?

Then try MX for `bath.edu`, an address the University uses for alumni (graduates and past staff)

DNS

Exercise Read about the public DNS servers like `1.1.1.1`, `8.8.8.8` and others; why would you want to use them over your local DNS server?

Exercise What are the privacy/security/legal aspects of using public servers?

Exercise Find out how to buy a DNS name

DNS

DNS requests are normally sent using UDP, and fit in a single UDP datagram

DNS

DNS requests are normally sent using UDP, and fit in a single UDP datagram

When the reply fits within 512 bytes, UDP is used for the reply by the server

DNS

DNS requests are normally sent using UDP, and fit in a single UDP datagram

When the reply fits within 512 bytes, UDP is used for the reply by the server

To keep datagrams small, if the reply is more than 512 bytes, the server sends a reply with a “truncated” flag set, and the client resends the request but using TCP

DNS

DNS requests are normally sent using UDP, and fit in a single UDP datagram

When the reply fits within 512 bytes, UDP is used for the reply by the server

To keep datagrams small, if the reply is more than 512 bytes, the server sends a reply with a “truncated” flag set, and the client resends the request but using TCP

Now, UDP is fast but unreliable, and 512 byte datagrams won’t be fragmented (recall: must be able to send 576 bytes IP), so there is little complication if a DNS datagram is lost: the source will just send a new request

DNS

UDP is preferred as it is fast with little overhead; while a TCP connection has a considerable setup cost

DNS

UDP is preferred as it is fast with little overhead; while a TCP connection has a considerable setup cost

So small and fast wherever possible; but slower and reliable if needed

DNS

DNS is good, but has problems

DNS

DNS is good, but has problems

- There is no security or authentication: if I get a reply saying that `138.38.32.14` is the address for `www.bath.ac.uk` can I trust this?

DNS

DNS is good, but has problems

- There is no security or authentication: if I get a reply saying that `138.38.32.14` is the address for `www.bath.ac.uk` can I trust this?
- Some server on the lookup path might have been subverted and made to hand out the wrong address; or a router may have carefully rewritten a DNS reply

DNS

DNS is good, but has problems

- There is no security or authentication: if I get a reply saying that `138.38.32.14` is the address for `www.bath.ac.uk` can I trust this?
- Some server on the lookup path might have been subverted and made to hand out the wrong address; or a router may have carefully rewritten a DNS reply
- Then I could receive a spoof IP address leading to someone else's web page: a problem if, say, I was looking for the page of a bank or shop

DNS

DNS is good, but has problems

- There is no security or authentication: if I get a reply saying that `138.38.32.14` is the address for `www.bath.ac.uk` can I trust this?
- Some server on the lookup path might have been subverted and made to hand out the wrong address; or a router may have carefully rewritten a DNS reply
- Then I could receive a spoof IP address leading to someone else's web page: a problem if, say, I was looking for the page of a bank or shop
- The spoof page could be made to look identical to the real bank web page, inviting me to enter my id and password

DNS

Exercise The security company RSA was attacked by DNS spoofing. Read about this

Exercise April 2018: routes to Amazon DNS servers were faked so that DNS requests were sent to fake servers. Read about this

Exercise Your home connection probably uses your ISP as a DNS server. ISPs have been known to rewrite DNS replies. They also might block access to other public DNS servers. Read about this

DNS

A partial solution exists in *Secure DNS* (DNSSec), which uses cryptography to authenticate DNS lookups

DNS

A partial solution exists in *Secure DNS* (DNSSec), which uses cryptography to authenticate DNS lookups

Not much in use as it was introduced when people still thought that cryptography was too expensive to use

DNS

And neither DNS nor DNSSec provide any privacy: anyone listening can see what hosts you are trying to contact by monitoring your DNS requests

DNS

And neither DNS nor DNSSec provide any privacy: anyone listening can see what hosts you are trying to contact by monitoring your DNS requests

So there has been a move by some people to use DNS over TLS (see TLS later and RFC7858) that encrypts DNS requests and replies

DNS

And neither DNS nor DNSSec provide any privacy: anyone listening can see what hosts you are trying to contact by monitoring your DNS requests

So there has been a move by some people to use DNS over TLS (see TLS later and RFC7858) that encrypts DNS requests and replies

This has a fair overhead over plain DNS, but provides both authentication and privacy

DNS

More surprisingly (at first), there is also DNS over HTTPS (DoH)

DNS

More surprisingly (at first), there is also DNS over HTTPS (DoH)

This sounds silly as HTTPS is a heavyweight protocol designed to move large Web pages, not lightweight DNS requests

DNS

More surprisingly (at first), there is also DNS over HTTPS (DoH)

This sounds silly as HTTPS is a heavyweight protocol designed to move large Web pages, not lightweight DNS requests

But:

DNS

More surprisingly (at first), there is also DNS over HTTPS (DoH)

This sounds silly as HTTPS is a heavyweight protocol designed to move large Web pages, not lightweight DNS requests

But:

- The overhead can be managed fairly well (see HTTP *Keep-Alive* and TLS reconnections)

DNS

More surprisingly (at first), there is also DNS over HTTPS (DoH)

This sounds silly as HTTPS is a heavyweight protocol designed to move large Web pages, not lightweight DNS requests

But:

- The overhead can be managed fairly well (see HTTP *Keep-Alive* and TLS reconnections)
- While an uncooperative ISP could block the DNS over TLS assigned port (853) the HTTPS port (443) cannot be blocked without a lot of collateral damage to normal browsing

DNS

Exercise CIDR makes PTR lookups harder as netmasks, and therefore the delegations, are no longer on a byte boundary. Read RFC2317 on how CNAMES are used to solve this problem

Exercise Read about how (or if) DoH is supported in your browser

Exercise Read about *DNS over Twitter* and *DNS over SMS*