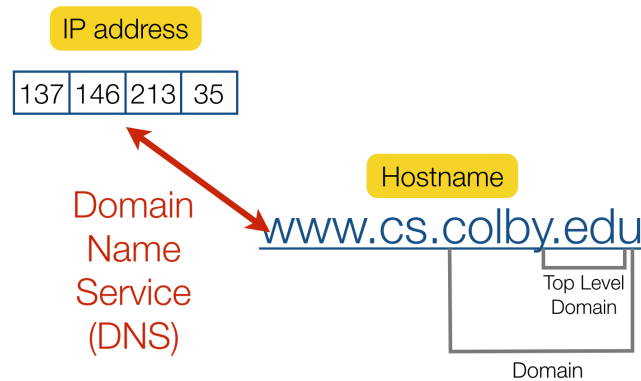


Application Layer (II)

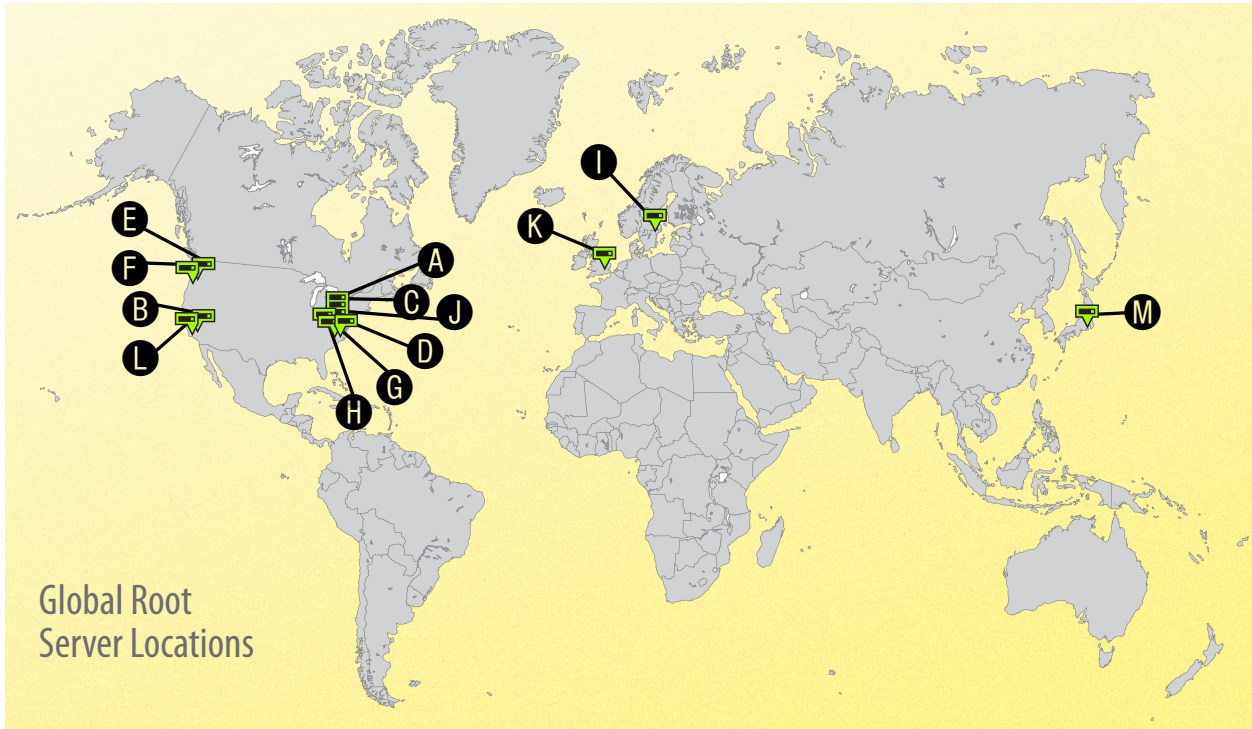
Domain Name Service



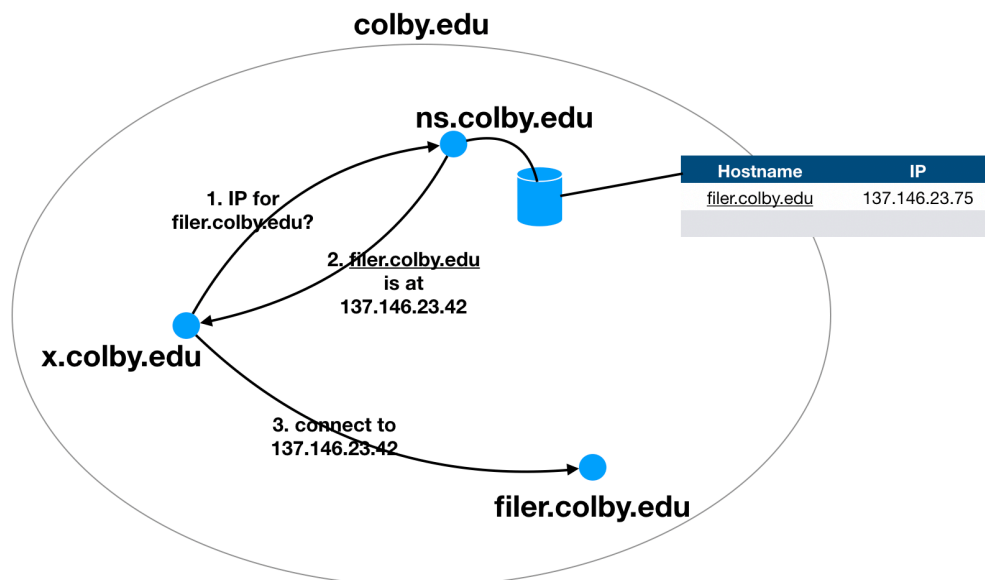
- The information contained in “www.cs.colby.edu” is three-fold:
 - Subdomain (cs)
 - Registered domain (colby)
 - Top-level domain (.edu)
 - Other top-level domains
 - Generic TLDs:
 - .com
 - .org
 - .net
 - .edu
 - .gov
 - ...
 - Country-code TLDs:
 - .uk
 - .cn
 - .jp
 - .ca
 - ...
- Domain name service provides the **mapping** between **hostnames** and **IP** addresses:
 - Is the mapping one-to-one, one-to-many, or many-to-one? [All are possible.]
 - one-to-one: cs.colby.edu
 - one-to-many: www.google.com [round robin]
 - many-to-one: Shared Web Hosting
 - A web hosting service where **many websites reside on one web server** connected to the Internet.

- [System administration is necessary](#). Benefit for users who do not want to deal with it, but hindrance to power users who want more control.
 - It is [cheaper](#) than dedicated server hosting.
 - Name-based virtual hosting: shared IP hosting, the virtual hosts serve multiple hostnames on a single machine with a single IP address.
 - HTTP requests carry the [target host/authority](#), allowing one server/IP address to serve multiple web names. With HTTPS, the TLS handshake also has to select the appropriate site/certificate, commonly using **SNI (Server Name Indication)**.
 - REF: https://en.wikipedia.org/wiki/Shared_web_hosting_service
- If you are asked to implement a system to provide domain name service, there are two possible solutions, centralized database and fully distributed database. Which solution you will select?
- If choosing the centralized one, the system has to face the possibility of a single point of failure, large traffic volume, distant centralized database, and maintenance challenges (as it's a huge database).
 - So, the fully distributed solution will be a better solution.
- The system to provide domain name service is actually a distributed redundant, hierarchical database.
- Root Name Servers
 - 13 logical root name servers are operated by 12 independent organizations
 - logical names in the form letter.root-servers.net, where letter ranges from a to m
 - each operator uses redundant computer equipment to provide reliable service
 - all (except B-root) operate in multiple geographical locations using anycast addressing
 - Top-level domain (TLD) servers
 - There is a TLD server cluster for each of the top-level domain (com, org, net, edu, and gov, and uk, fr, ca and jp)
 - Verisign Global Registry Services maintains the TLD servers for .com
 - Educause maintains the TLD servers for .edu
 - Authoritative DNS server
 - Every organization with publicly accessible hosts (e.g. web servers and mail servers) on the Internet must provide publicly accessible DNS records.
 - Implement own server or pay to have these records stored in an authoritative DNS server
 - Most universities and companies implement and maintain their own primary and secondary (backup) authoritative DNS servers.
 - Local DNS server
 - not strictly belong to the hierarchy
 - default name server, close to the host, may be in the same LAN
 - act as a proxy

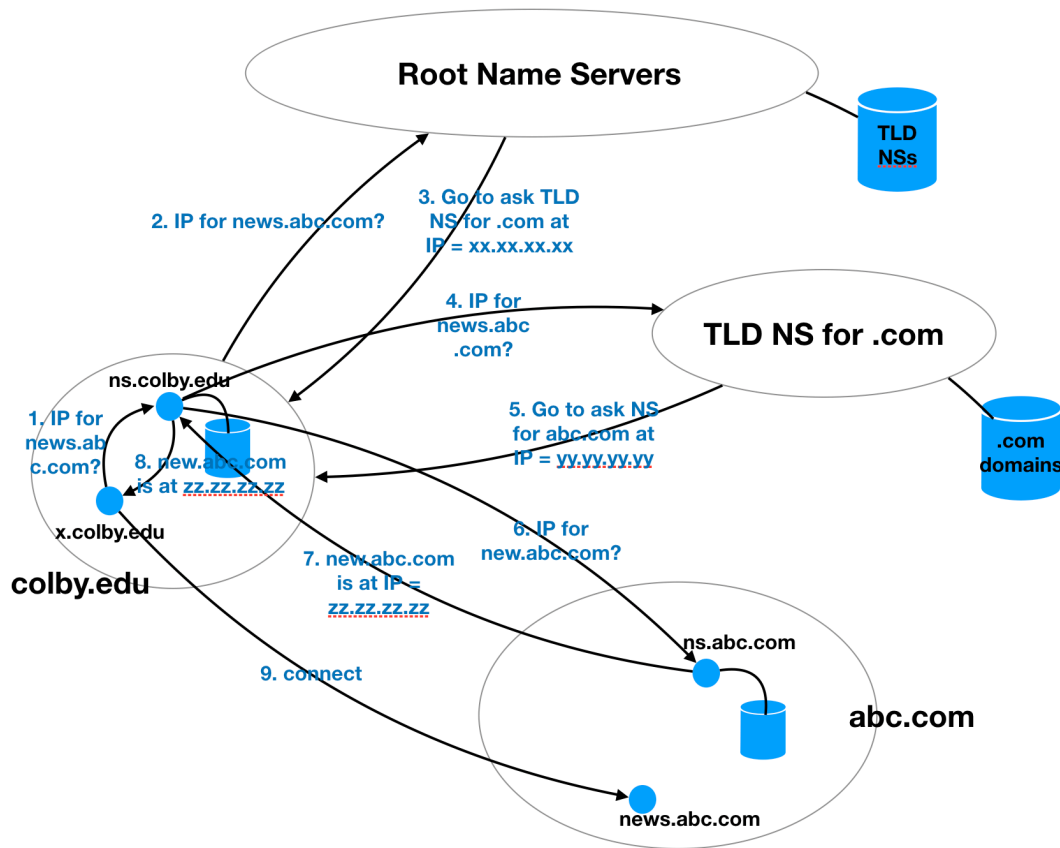
- can be used for DNS caching
- Query delegation
 - iterative (non-recursive): local domain name resolution
 - recursive: domain name resolution



- Example: Local Domain Name Resolution
 - If we are at Colby and want to connect to filer.colby.edu. This will lead to a local domain name resolution using Colby local DNS server.
 - ns.colby.edu (Colby local DNS server, 137.146.28.78) filer.colby.edu (137.146.23.75)



- Example: Non-local Domain Name Resolution
 - If we are at Colby and want to open abc.com. This will lead to a recursive domain name resolution.

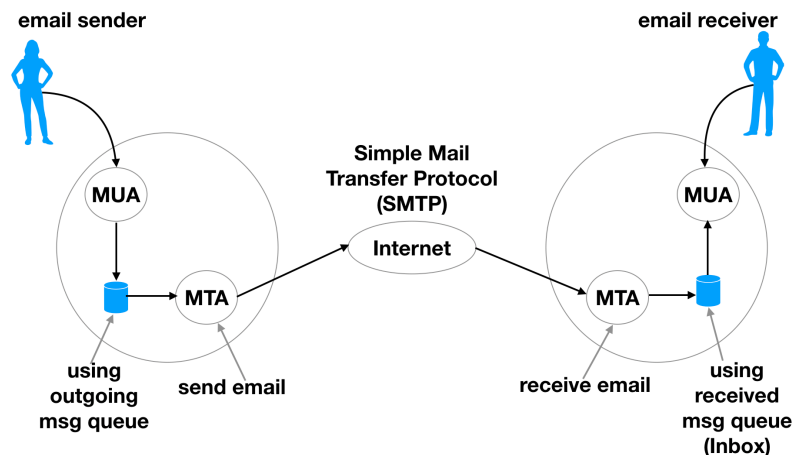


- Does this happen every time you visit `abc.com`?
 - No. A DNS record can be cached for a limited period (TTL).
 - Longer TTL leads to fewer DNS queries, but slower propagation of changes.
 - Shorter TTL leads to fresher information, but more DNS traffic.
- DNS tools:
 - `nslookup google.com`
 - `dig google.com` (A - v4, AAAA - v6, MX - mail exchange, NS - name server)
 - `whois google.com` (registration information)

Electronic Mail — SMTP

- Designed for **asynchronous** message delivery
 - This means that messages can be delivered to the other end of the network, and receivers can **read** the messages **when** they are **available**.
 - It **puts** a message **in a** message **pool** and **doesn't require immediate response**.
 - Robustness and reliability should also be considered in the design.

- Two types of agents
 - Message Transfer Agent (**MTA**)
 - A software **transfer** a **message** from one computer to another computer **using SMTP**
 - They are also called mail servers and often keep mailboxes for email.
 - Has two queues: outgoing message queue, received message queue (inbox)
 - Message User Agent (**MUA**):
 - A computer program allow users to read, reply to, forward, save, and compose message
 - They are **mail clients** and can be on the same host as MTA or not.



- From the above figure, we can observe that there are **two types of interactions** (and protocols) between these two types of agents
 - MTA to MTA (SMTP/ESMTP)
 - MUA to MTA (POP-3/IMAP)

MTA to MTA

- A protocol support MTA to MTA interaction is Simple Mail Transfer Protocol (**SMTP**): port 25
 - If the mailbox of the message recipient is not hosted locally, the message is forwarded to another MTA. Each relaying MAT adds a Received trace header field to the top of the message header.
 - SMTP is not built with no encryption nor authentication. Since the MUA was hosted on the same host as MTA, it could rely on the system authentication for user access.
- Therefore, a “not so simple” mail transfer protocol is needed. **ESMTP** (extended SMTP) addresses the authentication problem, which is adopted by many current emails, and **TLS**(Transport Layer Security) (port 587) is used to address encryption.

SMTP Example

- We use d122-01 to demonstrate a SMTP communication. In this example, we transmit a message from d122-01 to itself. The MUA is on the same host. We will use the program `mail` to check the received email.

```
d122-01:~> nc localhost 25
220 d122-01.colby.edu ESMTP Postfix (Ubuntu)
helo d122-01.colby.edu
250 d122-01.colby.edu
mail from: yingli@d122-01.colby.edu
250 2.1.0 Ok
rcpt to: yingli@d122-01.colby.edu
250 2.1.5 Ok
data
354 End data with <CR><LF>.<CR><LF>
subject: SMTP test
this is a test
.
250 2.0.0 Ok: queued as 1DD9C7C007F
quit
221 2.0.0 Bye

gloin:~> mail
Mail version 8.1.2 01/15/2001. Type ? for help.
"/var/mail/yingli": 1 message 1 new
>N 1 yingli@d122-01.c Mon Sep 30 12:27 14/552 SMTP test
& next
& quit
Saved 1 message in /personal/yingli/mbox
gloin:~>
```

- The red texts above are the responses from the mail server. The number in each response indicate the status of the mail server.
 - 220: The server is ready
 - 250 Ok: The prior request is completed
 - 354: The server is ready to receive data
 - 221: end the interaction
 - 550: you may receive 550 after rcpt, which means the address is not existed or it's returned by recipient's firewall.
- The green texts are the requests sent by users. The SMTP commands are:
 - helo: reach out the the server
 - mail from: followed by sender's address
 - rcpt to: followed by recipient's address
 - data: let the server know that you are going to send the message
 - subject: email subject
 - .: email body ends with a . on a newline by itself.
 - quit: end the interaction