

Domain Name System (DNS)



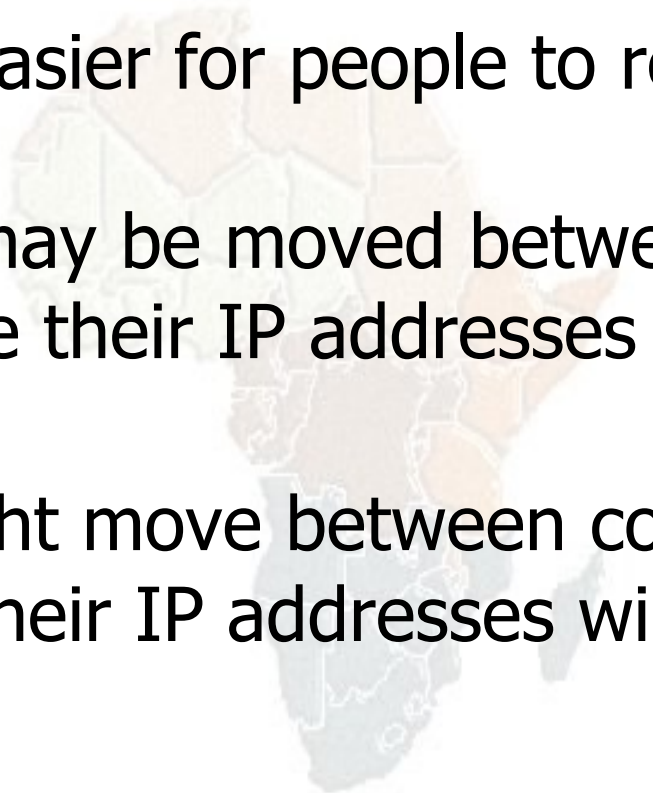
Session-1: Fundamentals

Michuki Mwangi

AfNOG Workshop, AIS 2018, Dakar

Computers use IP addresses.

Why do we need names?

- Names are easier for people to remember
 - Computers may be moved between networks, in which case their IP addresses will change.
 - Services might move between computers, in which case their IP addresses will change.
- 

The old solution: HOSTS.TXT

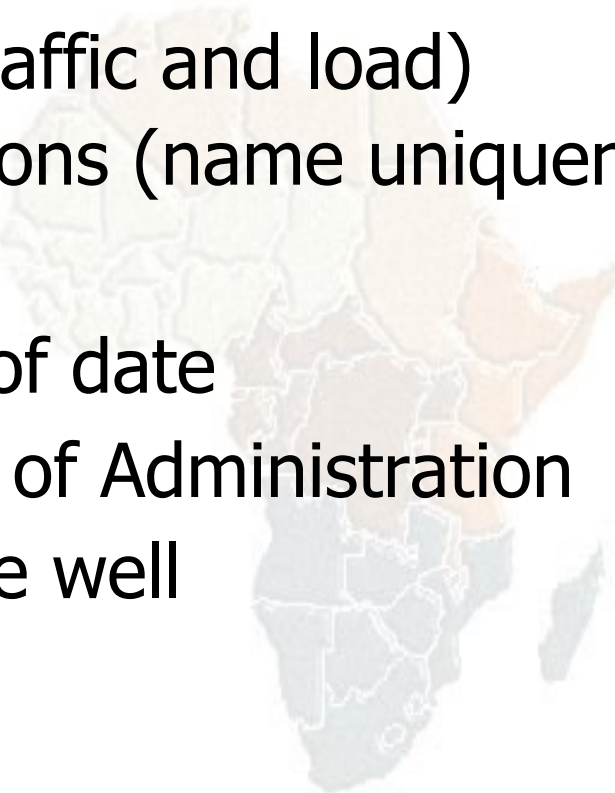
- A centrally-maintained file, distributed to all hosts on the Internet



- *SPARKY* 128.4.13.9
- *UCB-MAILGATE* 4.98.133.7
- *FTPHOST* 200.10.194.33
- ... etc

- This feature still exists:
 - */etc/hosts (UNIX)*
 - *c:\windows\hosts*

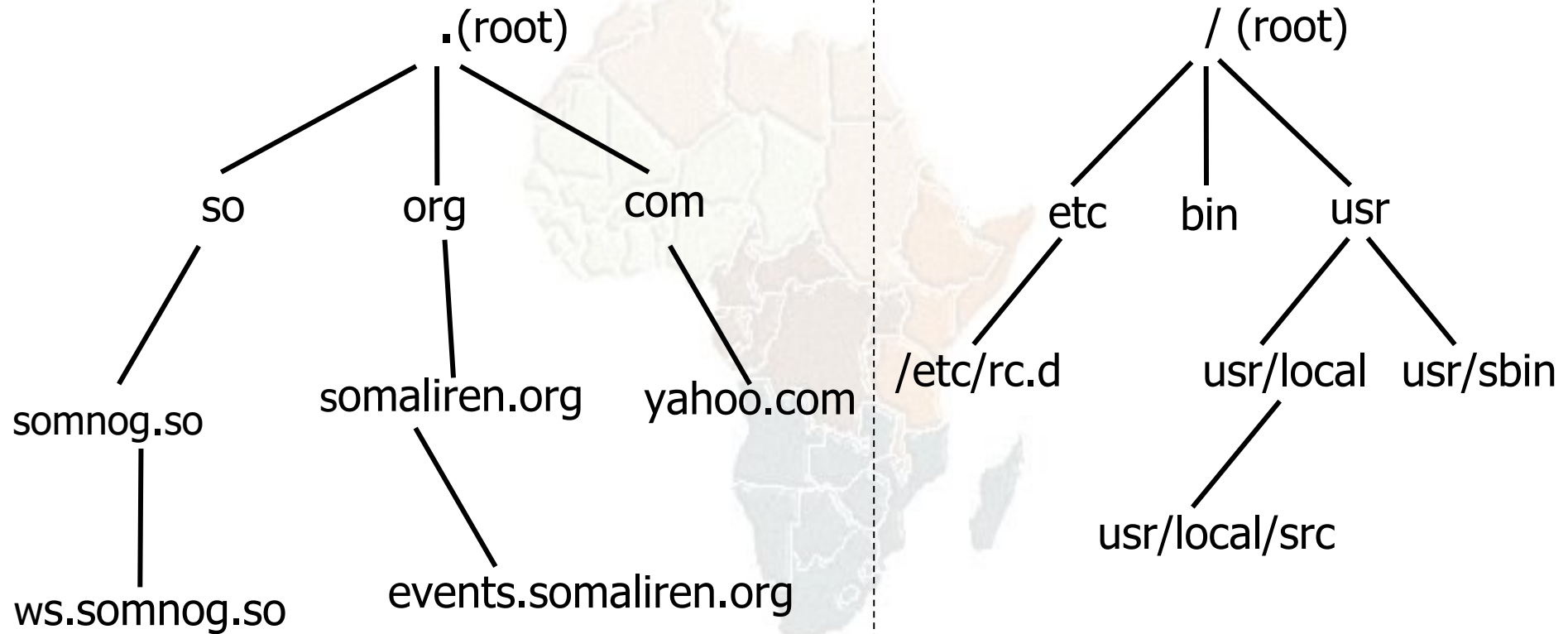
hosts.txt does not scale

- X Huge file (traffic and load)
 - X Name collisions (name uniqueness)
 - X Consistency
 - X Always out of date
 - X Single point of Administration
 - X Did not scale well
- 

The Domain Name System was born

- DNS is a distributed database for holding name to IP address (and other) information
- Distributed:
 - Shares the Administration
 - Shares the Load
- Robustness and improved performance achieved through
 - replication
 - and caching
- Employs a client-server architecture
- A critical piece of the Internet's infrastructure

DNS is Hierarchical



DNS Database

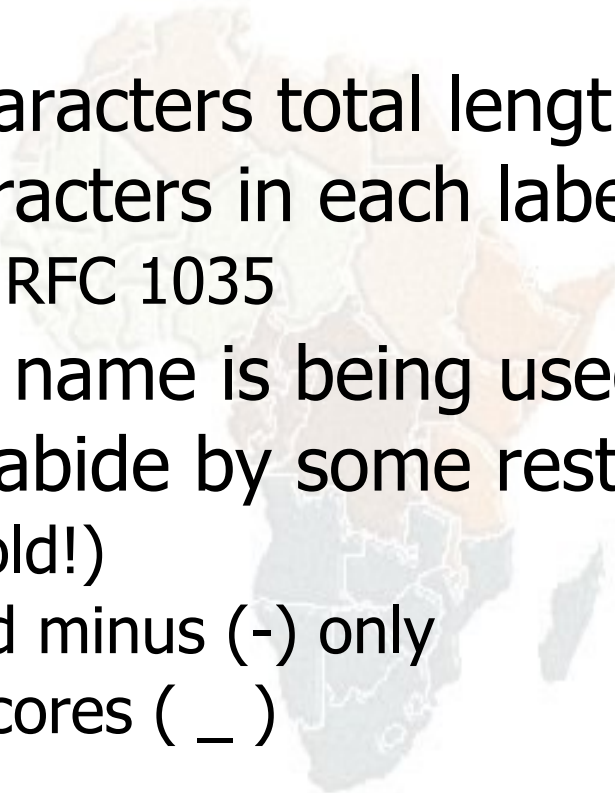
Unix Filesystem

Forms a tree structure

DNS is Hierarchical (contd.)

- Globally unique names
- Administered in zones (parts of the tree)
- You can give away ("delegate") control of part of the tree underneath you
- Example:
 - so on one set of nameservers
 - org.so on a different set
 - somaliren.org.so on a different set
 - event.somaliren.org.so on a different set

Domain Names are (almost) unlimited

- Max 255 characters total length
 - Max 63 characters in each label
 - RFC 1034, RFC 1035
 - If a domain name is being used as a host name, you should abide by some restrictions
 - RFC 952 (old!)
 - a-z 0-9 and minus (-) only
 - No underscores (_)
- 

Using the DNS

- A Domain Name (like `events.somaliren.org.so`) is the KEY to look up information
- The result is zero or more RESOURCE RECORD SETS (RRSets)
- There are different RRTYPES for different types of information
- You can ask for the specific type you want, or ask for "any" RRs associated with the domain name

Commonly seen Resource Record Types (RRTYPEs)

- A (address): map hostname to IPv4 address
- AAAA (quad A): map a hostname to IPv6 address
- PTR (pointer): map IP address to hostname
- MX (mail exchanger): where to deliver mail for a mail domain
- CNAME (canonical name): map alternative hostname to real hostname
- TXT (text): any descriptive text
- NS (name server), SOA (start of authority): used for delegation and management of the DNS itself

A Simple Example

- Query: **somnog.so.**
- Query type: **A**
- Result:

somnog.so. 14400 IN A 216.172.184.215

- In this case a single RR is found, but a set of multiple RRs may be returned.
 - (IN is the "class" for INTERNET use of the DNS)

Possible results from a Query

- POSITIVE ("NOERROR")
 - the name exists, and has zero or more RRSets associated with it
- NEGATIVE ("NXDOMAIN")
 - the name does not exist
- SERVER FAILURE ("SERVFAIL")
 - server is having bad hair day
- FORMAT ERROR ("FORMERR")
 - the query you sent was broken in some way
- REFUSED ("REFUSED")
 - You are not allowed to query the server

How do you use an IP address as the key for a DNS query

- Convert the IP address to dotted-quad
- Reverse the four parts
- Add ".in-addr.arpa." to the end; special domain reserved for this purpose

e.g. to find name for 193.194.185.25

Domain name: 25.185.194.193.in-addr.arpa.

Query Type: PTR

Result: ashanti.gh.com.

Known as a "reverse DNS lookup" (because we are looking up the name for an IP address, rather than the IP address for a name)

Any Questions?



DNS is a Client-Server application

- (Of course - it runs across a network)
- Requests and responses are most frequently carried in UDP packets, port 53
- DNS can also use TCP transport, port 53
 - for large responses (not just zone transfers)
 - because you want to exchange more than one query/response on a single session
 - because a response you received told you to wait with TC=1
 - because UDP is being aggressively rate-limited or blocked (e.g. to mitigate a reflection attack)

There are three roles involved in DNS

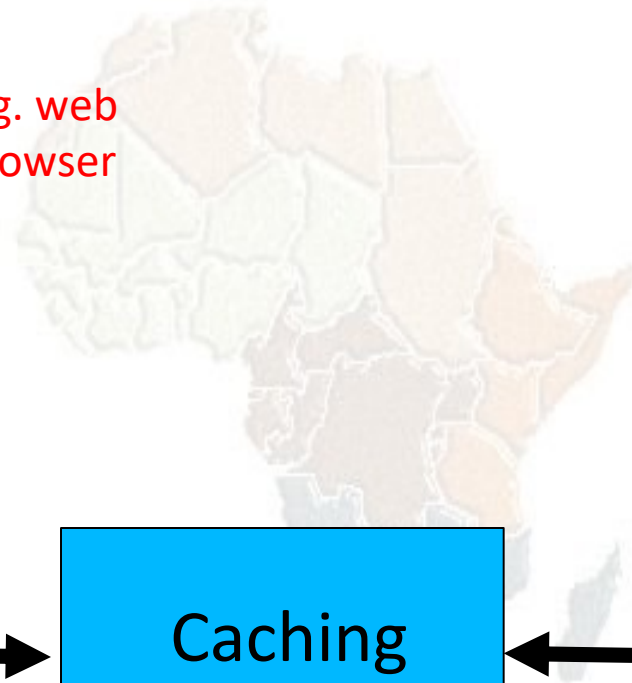
Application

e.g. web
browser

Stub
Resolver

Caching
Resolver

Authoritative
Nameserver



Three roles in DNS

- **STUB RESOLVER**
 - Takes request from application, formats it into UDP packet, sends to recursive resolver
- **RECURSIVE RESOLVER**
 - Returns the answer if already known
 - Otherwise searches for an authoritative server which has the information
 - Caches the result for future queries
- **AUTHORITATIVE NAMESERVER**
 - Contains the actual information published in the DNS by the domain owner

Three roles in DNS

- The SAME protocol is used for stub<-> resolver and resolver <-> auth nameserver
- It is possible to configure a single name server as both a resolver and an authoritative server
- But it still performs only one role for each incoming query
- Common but **NOT RECOMMENDED** to configure in this way (we will see why later).

ROLE 1: THE STUB RESOLVER

- A piece of software which formats a DNS request into a DNS message, sends it to a resolver, and decodes the response when it arrives
- Usually a shared library (e.g. libresolv.so under Unix) because so many applications need it
- EVERY host needs a stub resolver - e.g. every Windows workstation has one

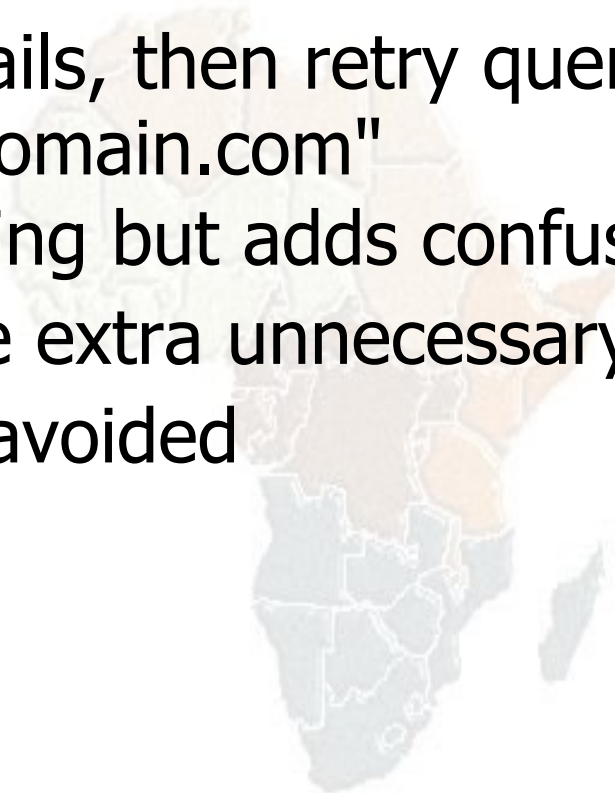
How does the stub resolver find a recursive nameserver?

- It has to be explicitly configured (statically, or via DHCP etc)
- Must be configured with the IP ADDRESS of a cache (why not its name?)
- Good idea to configure more than one recursive nameserver, in case the first one fails
 - But failover between them might not be quick

How do you choose which recursive resolver(s) to configure?

- Must have PERMISSION to use it
 - e.g. recursive resolver at your ISP, or your own, or a deliberately-public one
- Prefer a nearby recursive resolver
 - Minimises round-trip time and packet loss
 - Can reduce traffic on your external link, since often the cache can answer without contacting other servers
- Prefer a reliable recursive resolver
 - Perhaps your own?

Stub resolvers can be configured with search domain(s)

- If "foo.bar" fails, then retry query as "foo.bar.mydomain.com"
 - Can save typing but adds confusion
 - May generate extra unnecessary traffic
 - Usually best avoided
- 

Example: Unix stub resolver configuration

/etc/resolv.conf

```
search sse.ws.afnog.org  
nameserver 196.200.223.10  
nameserver 196.200.223.12
```

That's all you need to configure a resolver

Testing DNS

- Just put "www.yahoo.com" in a web browser?
- Why is this not a good test?



Testing DNS with "dig"

- "dig" is a program which just makes DNS queries and displays the results
- Better than "nslookup", "host" because it shows the raw information in full

```
dig events.somaliren.org.so.
```

```
-- defaults to query type "A"
```

```
dig somaliren.org. mx
```

```
-- specified query type
```

```
dig @154.73.25.70 somaliren.org.so a
```

```
-- send to particular cache (overrides  
/etc/resolv.conf)
```

The trailing dot

```
# dig ws.afnog.org.
```

A red arrow points from the right side of the slide towards the trailing dot at the end of the command 'dig ws.afnog.org.' in the code block above.

- Prevents any default domain being appended
- Get into the habit of using it always when testing DNS
 - only on domain names, not IP addresses or e-mail addresses

```
[field@term /usr/home/field]$ dig @zoe.dns.gh. downloads.dns.gh. a
```

```
; <<>> DiG 9.3.1 <<>> @zoe.dns.gh. downloads.dns.gh. a
```

```
; (1 server found)
```

```
;; global options: printcmd
```

```
;; Got answer:
```

```
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 34963
```

```
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 2, AUTHORITY: 3, ADDITIONAL: 0
```

```
;; QUESTION SECTION:
```

```
;downloads.dns.gh. IN A
```

```
;; ANSWER SECTION:
```

```
downloads.dns.gh. 3600 IN CNAME zoe.dns.gh.
```

```
zoe.dns.gh. 3600 IN A 147.28.0.23
```

```
;; AUTHORITY SECTION:
```

```
dns.gh. 3600 IN NS zoe.dns.gh.
```

```
dns.gh. 3600 IN NS mantse.gh.com.
```

```
dns.gh. 3600 IN NS snshq902.ghanatel.com.gh.
```

```
;; Query time: 275 msec
```

```
;; SERVER: 147.28.0.23#53(147.28.0.23)
```

```
;; WHEN: Sat May 24 00:17:53 2008
```

```
;; MSG SIZE rcvd: 145
```

Understanding output from dig

- **STATUS**
 - NOERROR: 0 or more RRs returned
 - NXDOMAIN: non-existent domain
 - SERVFAIL: cache could not locate answer
 - REFUSED: query not available on cache server
- **FLAGS**
 - AA: Authoritative answer (not from cache)
 - You can ignore the others
 - QR: Query/Response (1 = Response)
 - RD: Recursion Desired
 - RA: Recursion Available
- **ANSWER:** number of RRs in answer

Understanding output from dig

- Answer section (RRs requested)
 - Each record has a Time To Live (TTL)
 - Says how long the cache will keep it
- Authority section
 - Which nameservers are authoritative for this domain
- Additional section
 - More RRs (typically IP addresses for the authoritative nameservers)
- Total query time
- Check which server gave the response!
 - If you make a typing error, the query may go to a default server