ADDIS ABABA INSTITUTE OF TECHNOLOGY
አዲስ አበባ ቴክኖሎጂ ኢንስቲትዩት
ADDIS ABABA UNIVERSITY
አዲስ አበባ ዩኒቨርሲቲ

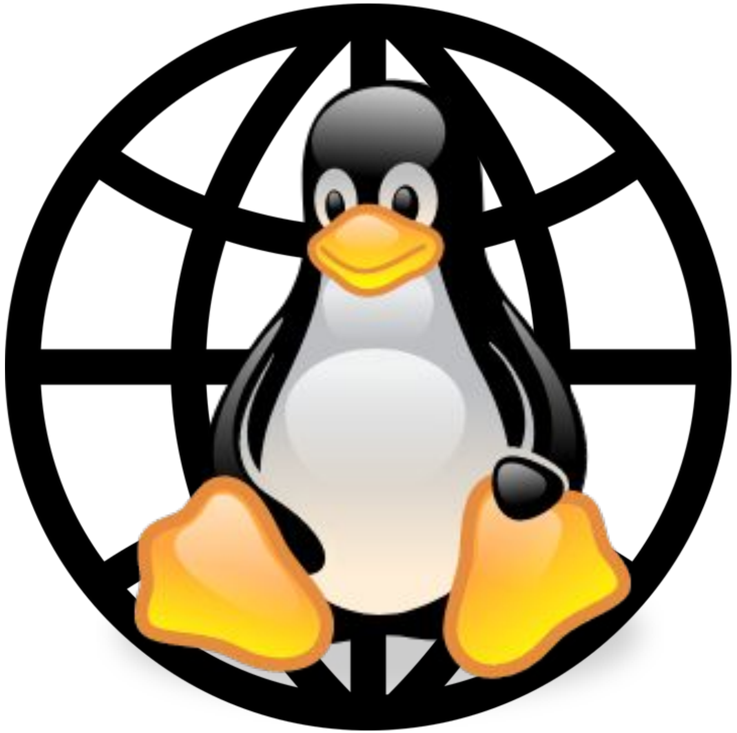
Enterprise Systems and Network Administration

CHAPTER FOUR

Web and Domain Name System

Getnet M.
getnet.mamo@aau.edu.et

Web and Domain Name System



Introduction

- Web server
  - Receive request from clients using URL
  - Translate a URL either into a filename or program/script to run
  - Send back the file or result of executed program/script
- URL stands for Uniform Resource Locator.
  - http://www.aau.edu.et/images/president.jpg
  - comes in three parts:<proto>://<host>/<path>

Introduction

- URL
  - Proto: http, https, ftp, or others
  - Host: www.aau.edu.et or IP address
  - Path: /images/president.jpg
- Web browser has to send requests using commands that can be handled by the web-server:
  - GET / HTTP/1.1
  - Other methods: POST, CONNECT, DELETE, PUT
- Web server has to listen to those requests and respond accordingly

Introduction

- HOW TO CHOOSE WEB-SERVER?
  - Performance: run faster with minimum hardware
  - Supported features
  - Error messages and appropriate response
  - Supported scripting languages and modules
  - Scalability and cluster
  - Security
  - Platform
  - Cost
  - Support

Introduction

- Why Apache?
- Apache is popular:
  - Stable
  - Used by major companies like amazon
  - Open-source
  - Runs on most platforms (Linux,windows,Unix.)
  - Extremely flexible
  - Secured

Introduction

- **Apache: HTTP server**
- **Firefox, Chromium, IE, Opera: HTTP client**
- **Client:**
  - establishes TCP connection on port 80
  - Requests server by sending HTTP commands
- **Server**
  - Accepts connection from clients
  - Receives commands and sends reply to clients

Web and HTTP

First some jargon

- Web page consists of objects
- Object can be HTML file, JPEG image, Java applet, audio file,…
- Web page consists of base HTML-file which includes several referenced objects
- Each object is addressable by a URL (Uniform Resource Locator)
- URL:

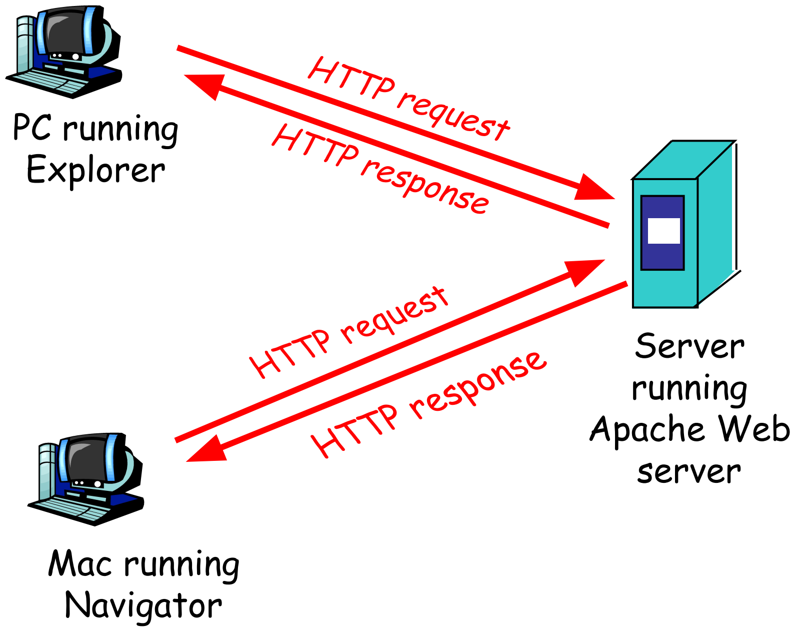
$$\underbrace{\texttt{www.site.aau.edu.et}}_{\text{host name}}\texttt{/}\underbrace{\texttt{dept/pic.gif}}_{\text{path name}}$$

HTTP Protocol

HTTP: hypertext transfer protocol

Web's application layer protocol

- client/server model
  - *client:* browser that requests, receives, "displays" Web objects
  - *server:* Web server sends objects in response to requests
- HTTP 1.0: RFC 1945
- HTTP 1.1: RFC 2068
- HTTP 2: RFC 9113
- HTTP 3: RFC 9114 (Latest)



HTTP Protocol

Uses TCP:

- client initiates TCP connection (creates socket) to server, port 80
- server accepts TCP connection from client
- HTTP messages (application-layer protocol messages) exchanged between browser (HTTP client) and Web server (HTTP server)
- TCP connection closed

HTTP is "stateless"

- server maintains no information about past client requests

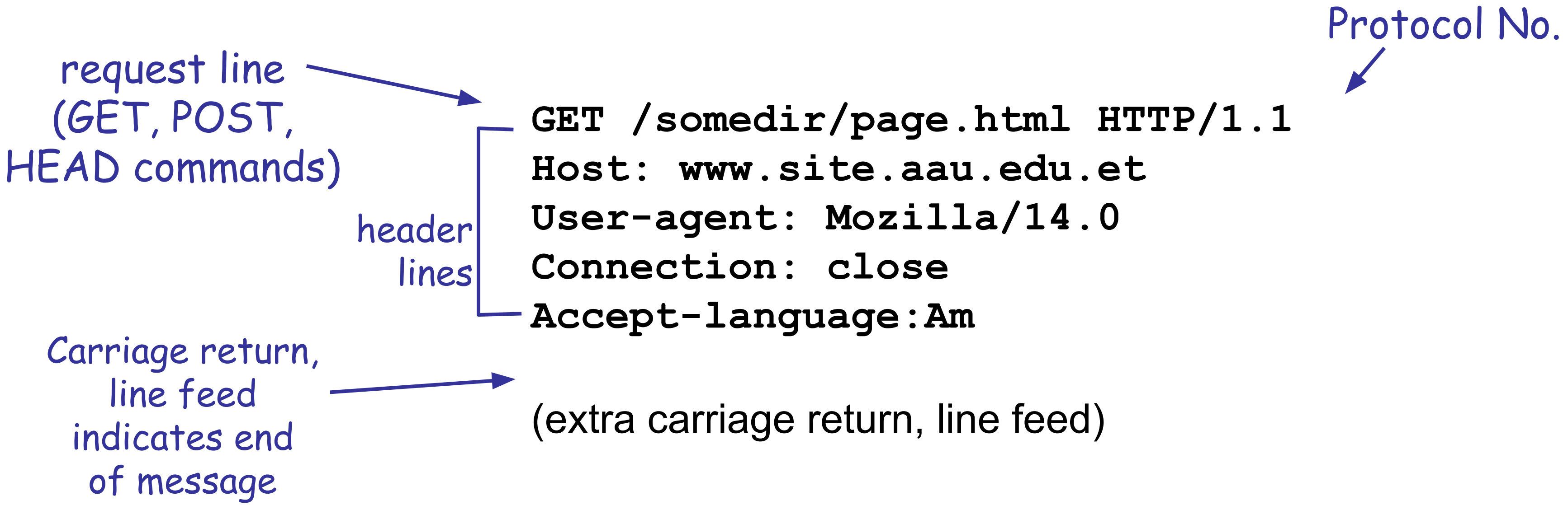
aside

Protocols that maintain "state" are complex!

- past history (state) must be maintained
- if server/client crashes, their views of "state" may be inconsistent, must be reconciled

HTTP Request Message

- two types of HTTP messages: *request*, *response*
- HTTP request message:
  - ASCII (human-readable format)



request line (GET, POST, HEAD commands)
Protocol No.
header lines
Carriage return, line feed indicates end of message

```
GET /somedir/page.html HTTP/1.1
Host: www.site.aau.edu.et
User-agent: Mozilla/14.0
Connection: close
Accept-language:Am

(extra carriage return, line feed)
```

Uploading Form Input

Post method:

- Uses POST method
- Web page often includes form input
- Input content is uploaded to server in "entity body" in request message

URL method:

- Uses GET method
- Input is uploaded in URL field of request line:

```
www.somesite.com/animalsearch?monkeys&banana
```

What are the other methods?what is their function? How do they work?

HTTP Response Message

status line (protocol status code status phrase)

```
HTTP/1.1 200 OK
```

header lines

```
Connection close
Date: Thu, 06 Aug 2022 12:00:15 GMT
Server: Apache/1.3.0 (Unix)
Last-Modified: Mon, 22 Jun 2022 …...
Content-Length: 6821
Content-Type: text/html
```

data, e.g., requested HTML file, image

```
data data data data data ...
```

HTTP Response Status Codes

In first line in server->client response message.

A few sample codes:

200 OK

- request succeeded, requested object later in this message

301 Moved Permanently

- requested object moved, new location specified later in this message (Location:) one way of URL redirection

400 Bad Request

- request message not understood by server

404 Not Found

- requested document not found on this server

505 HTTP Version Not Supported

Trying HTTP on a terminal

1. Telnet to your favorite Web server:

```
telnet www.mom.gov.et 80
```

Opens TCP connection to port 80 (default HTTP server port) at www.mom.gov.et
Anything typed in sent to port 80 at www.mom.gov.et

2. Type in a GET HTTP request:

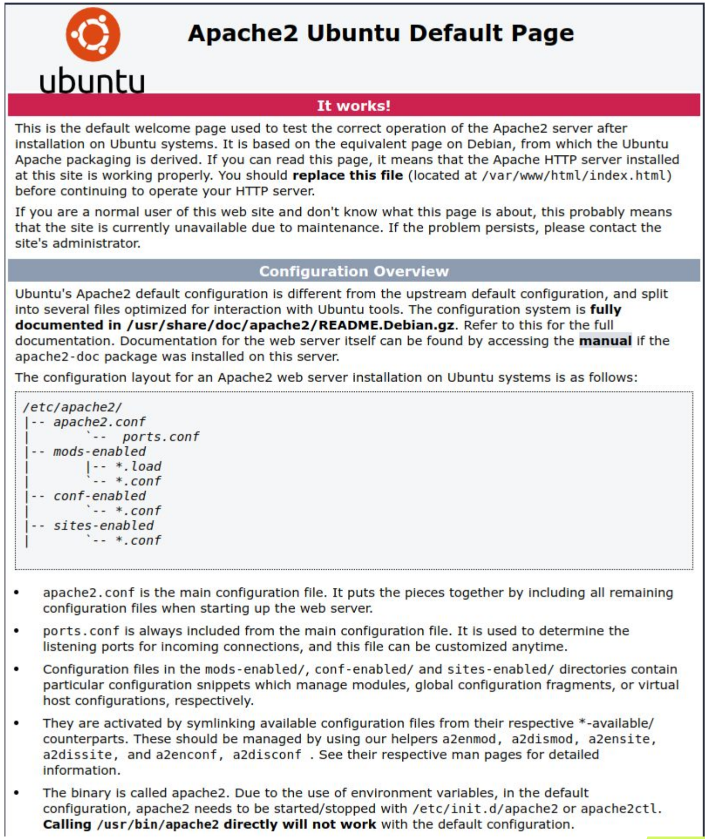
```
GET / HTTP/1.1
Host: www.mom.gov.et
```

By typing this in (hit carriage return twice), you send this minimal (but complete) GET request to HTTP server

3. Look at response message sent by HTTP server!

Apache Web Server

- Ubuntu/Debian distributions
  - *apt-get install apache2*
- Daemon starts automatically after installation
- Apache comes with several modules
- Apache is extensible by installing modules
- Some examples of modules:
  - *mod_cgi, mod_perl, mod_aspdotnet, mod_ssl, mode_ftpd*
- You can also develop your own module:
  - Using the Apache module API documentation
  - https://httpd.apache.org/docs/2.4/mod/



Apache2 Ubuntu Default Page

ubuntu

It works!

This is the default welcome page used to test the correct operation of the Apache2 server after installation on Ubuntu systems. It is based on the equivalent page on Debian, from which the Ubuntu Apache packaging is derived. If you can read this page, it means that the Apache HTTP server installed at this site is working properly. You should **replace this file** (located at `/var/www/html/index.html`) before continuing to operate your HTTP server.

If you are a normal user of this web site and don't know what this page is about, this probably means that the site is currently unavailable due to maintenance. If the problem persists, please contact the site's administrator.

Configuration Overview

Ubuntu's Apache2 default configuration is different from the upstream default configuration, and split into several files optimized for interaction with Ubuntu tools. The configuration system is **fully documented in /usr/share/doc/apache2/README.Debian.gz**. Refer to this for the full documentation. Documentation for the web server itself can be found by accessing the **manual** if the apache2-doc package was installed on this server.

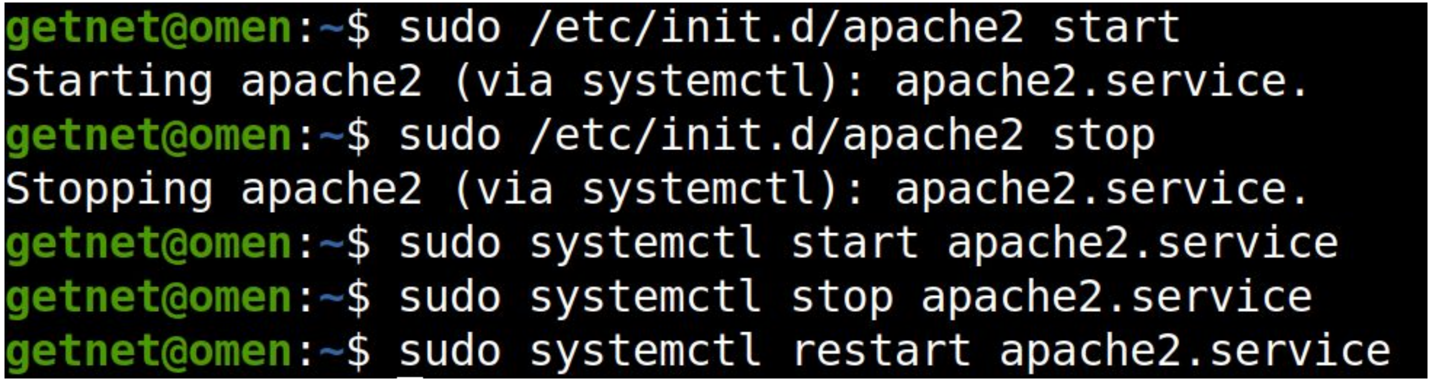
The configuration layout for an Apache2 web server installation on Ubuntu systems is as follows:

```
/etc/apache2/
|-- apache2.conf
|       `--  ports.conf
|-- mods-enabled
|       |-- *.load
|       `-- *.conf
|-- conf-enabled
|       `-- *.conf
|-- sites-enabled
|       `-- *.conf
```

- `apache2.conf` is the main configuration file. It puts the pieces together by including all remaining configuration files when starting up the web server.
- `ports.conf` is always included from the main configuration file. It is used to determine the listening ports for incoming connections, and this file can be customized anytime.
- Configuration files in the `mods-enabled/`, `conf-enabled/` and `sites-enabled/` directories contain particular configuration snippets which manage modules, global configuration fragments, or virtual host configurations, respectively.
- They are activated by symlinking available configuration files from their respective *-available/ counterparts. These should be managed by using our helpers `a2enmod`, `a2dismod`, `a2ensite`, `a2dissite`, and `a2enconf`, `a2disconf` . See their respective man pages for detailed information.
- The binary is called apache2. Due to the use of environment variables, in the default configuration, apache2 needs to be started/stopped with `/etc/init.d/apache2` or `apache2ctl`. **Calling `/usr/bin/apache2` directly will not work** with the default configuration.

Starting and Shutting Apache2

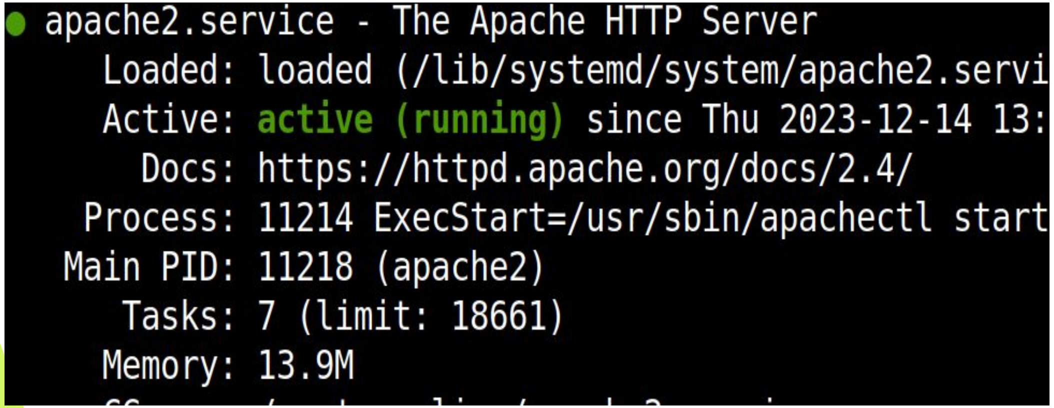
- start apache
  - *sudo /etc/init.d/apache2 start*
- To stop apache
  - *sudo /etc/init.d/apache2 stop*
- Apache should be configured to start at boot time



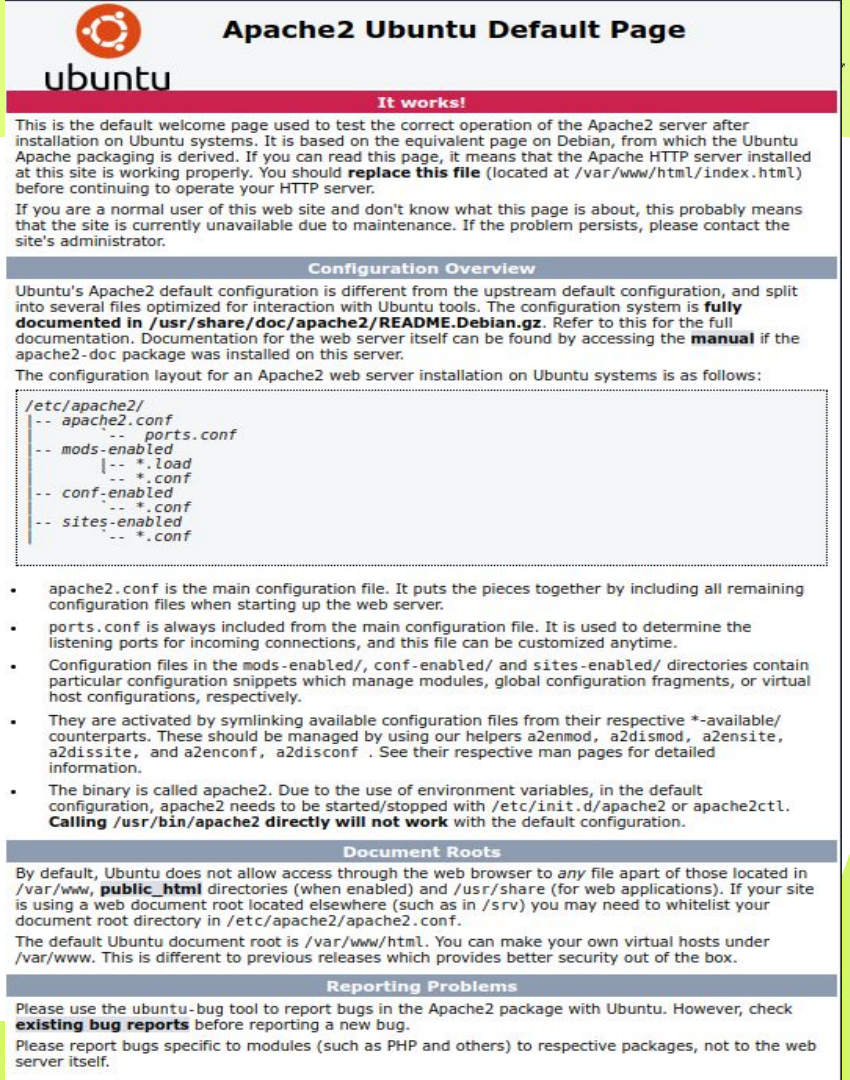
```
getnet@omen:~$ sudo /etc/init.d/apache2 start
Starting apache2 (via systemctl): apache2.service.
getnet@omen:~$ sudo /etc/init.d/apache2 stop
Stopping apache2 (via systemctl): apache2.service.
getnet@omen:~$ sudo systemctl start apache2.service
getnet@omen:~$ sudo systemctl stop apache2.service
getnet@omen:~$ sudo systemctl restart apache2.service
```

Testing Apache

- Browse to localhost or IP of server
  - You should get *IT WORKS!* Default page
- To check the service
  - *sudo systemctl status apache2.service*



```
● apache2.service - The Apache HTTP Server
     Loaded: loaded (/lib/systemd/system/apache2.servi
     Active: active (running) since Thu 2023-12-14 13:
       Docs: https://httpd.apache.org/docs/2.4/
    Process: 11214 ExecStart=/usr/sbin/apachectl start
   Main PID: 11218 (apache2)
      Tasks: 7 (limit: 18661)
     Memory: 13.9M
```



Apache2 Ubuntu Default Page

ubuntu

It works!

This is the default welcome page used to test the correct operation of the Apache2 server after installation on Ubuntu systems. It is based on the equivalent page on Debian, from which the Ubuntu Apache packaging is derived. If you can read this page, it means that the Apache HTTP server installed at this site is working properly. You should **replace this file** (located at /var/www/html/index.html) before continuing to operate your HTTP server.

If you are a normal user of this web site and don't know what this page is about, this probably means that the site is currently unavailable due to maintenance. If the problem persists, please contact the site's administrator.

Configuration Overview

Ubuntu's Apache2 default configuration is different from the upstream default configuration, and split into several files optimized for interaction with Ubuntu tools. The configuration system is **fully documented in /usr/share/doc/apache2/README.Debian.gz**. Refer to this for the full documentation. Documentation for the web server itself can be found by accessing the **manual** if the apache2-doc package was installed on this server.

The configuration layout for an Apache2 web server installation on Ubuntu systems is as follows:

```
/etc/apache2/
|-- apache2.conf
|       `--  ports.conf
|-- mods-enabled
|       |-- *.load
|       `-- *.conf
|-- conf-enabled
|       `-- *.conf
|-- sites-enabled
|       `-- *.conf
```

- apache2.conf is the main configuration file. It puts the pieces together by including all remaining configuration files when starting up the web server.
- ports.conf is always included from the main configuration file. It is used to determine the listening ports for incoming connections, and this file can be customized anytime.
- Configuration files in the mods-enabled/, conf-enabled/ and sites-enabled/ directories contain particular configuration snippets which manage modules, global configuration fragments, or virtual host configurations, respectively.
- They are activated by symlinking available configuration files from their respective *-available/ counterparts. These should be managed by using our helpers a2enmod, a2dismod, a2ensite, a2dissite, and a2enconf, a2disconf . See their respective man pages for detailed information.
- The binary is called apache2. Due to the use of environment variables, in the default configuration, apache2 needs to be started/stopped with /etc/init.d/apache2 or apache2ctl. **Calling /usr/bin/apache2 directly will not work** with the default configuration.

Document Roots

By default, Ubuntu does not allow access through the web browser to *any* file apart of those located in /var/www, **public_html** directories (when enabled) and /usr/share (for web applications). If your site is using a web document root located elsewhere (such as in /srv) you may need to whitelist your document root directory in /etc/apache2/apache2.conf.

The default Ubuntu document root is /var/www/html. You can make your own virtual hosts under /var/www. This is different to previous releases which provides better security out of the box.

Reporting Problems

Please use the ubuntu-bug tool to report bugs in the Apache2 package with Ubuntu. However, check **existing bug reports** before reporting a new bug.

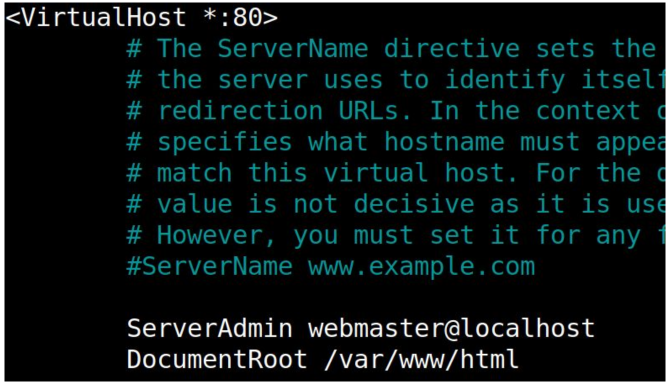
Please report bugs specific to modules (such as PHP and others) to respective packages, not to the web server itself.

Configure Apache

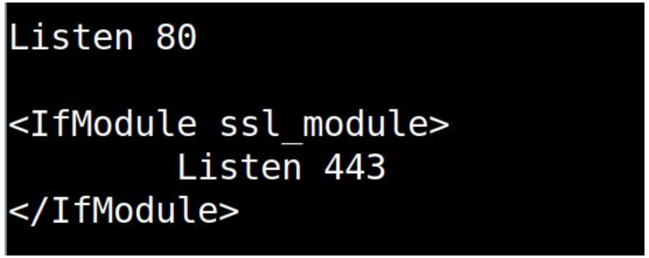
- Default apache configuration is usually quite good
- Customization is possible
- To change the default welcome page put *index.html* or *index.php* under */var/www/html/*
- To make the file world readable change the permission:
  - *chmod 644 index.html*
  - *chown www-data:www-data /var/www/index.html*
- The main configuration file in debian/ubuntu is
  - */etc/apache2/apache2.conf*

Configure Apache

- **Configuration options:**
  - ServerRoot: server root directory, /etc/apache2
  - Listen: specify listening port, default 80
  - ServerName: specify FQDN
  - DocumentRoot: /var/www/
  - MaxClients: max number of concurrent clients
  - LoadModule: adding or loading modules
  - Include: include other configuration files
  - VirtualHost: allows a server to host multiple sites

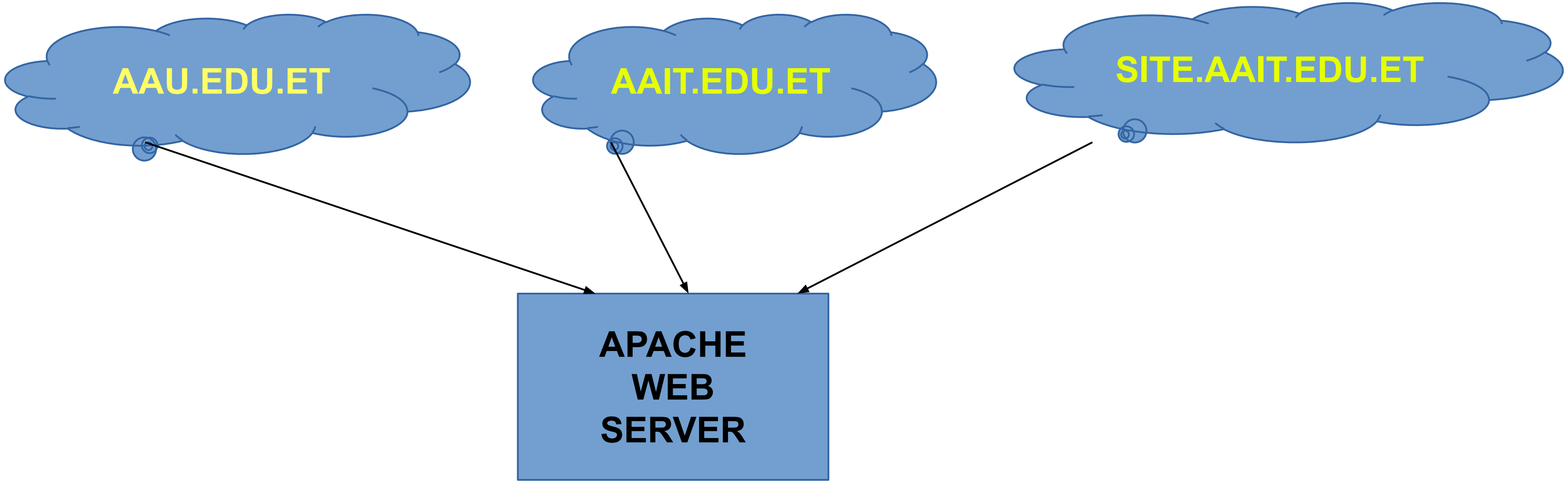


```
<VirtualHost *:80>
        # The ServerName directive sets the
        # the server uses to identify itsel
        # redirection URLs. In the context o
        # specifies what hostname must appea
        # match this virtual host. For the d
        # value is not decisive as it is use
        # However, you must set it for any f
        #ServerName www.example.com

        ServerAdmin webmaster@localhost
        DocumentRoot /var/www/html
```



```
Listen 80

<IfModule ssl_module>
        Listen 443
</IfModule>
```

Virtual Hosting



Virtual Host: Ability of web-server to host multiple domains or virtual hosts

Virtual Hosting

```
<VirtualHost www.aau.edu.et:80>
    ServerAdmin webadmin@aau.edu.et
    DocumentRoot /var/www/aau/
    ServerName www.aau.edu.et
    ErrorLog /var/logs/aau
</VirtualHost>
```

```
<VirtualHost www.aait.edu.et:80>
    ServerAdmin webadmin@aait.edu.et
    DocumentRoot /var/www/aait/
    ServerName www.aait.edu.et
    ErrorLog /var/logs/aait
</VirtualHost>
```

Virtual Hosting

- The value of the ***ServerName*** option in the ***VirtualHost*** container must be a name that is resolvable via DNS to the web **server machine IP** address
- Each ***DocumentRoot*** should point to the root of the complete website of virtual host.
- To add multiple virtual hosts it needs to add the ***virutalHost*** tag as many as number of virtual hosts.
- Server should be restarted after adding a new virtual host.

Domain Name Systems

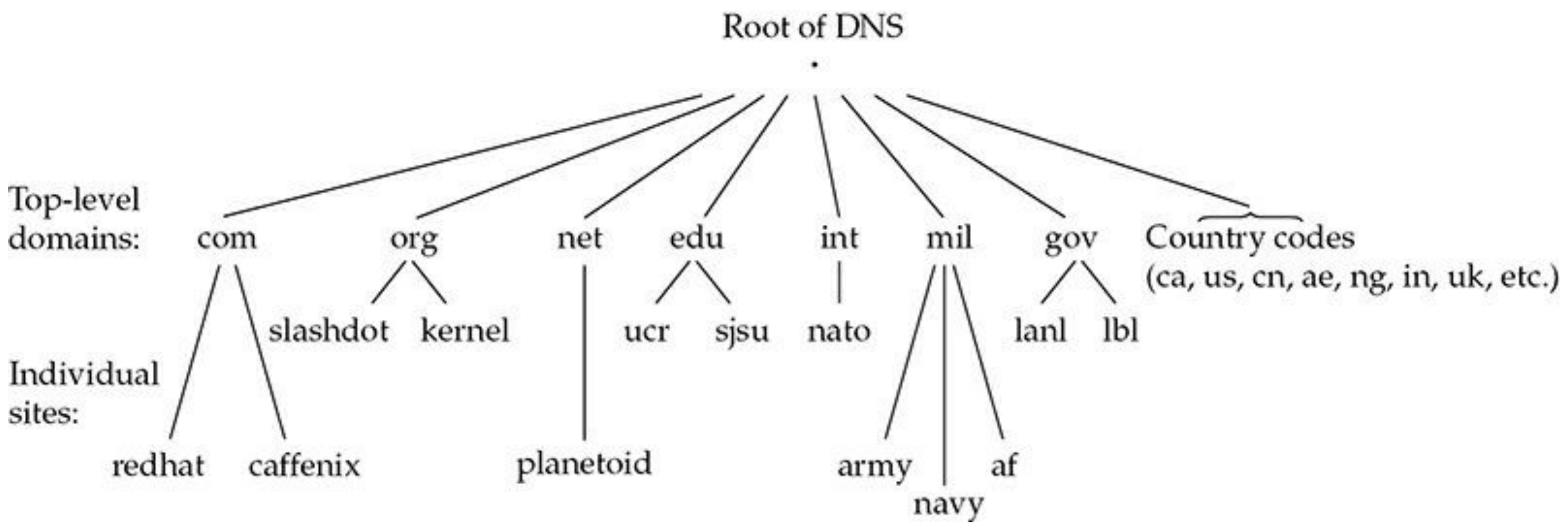
Domain Name System

- Most sites referenced by their fully qualified domain name (FQDN),like this one:
  - **www.aau.edu.et**
- Each string between the periods in this FQDN is significant.
- Starting from the right and moving to the left, you have the top-level domain component, the second-level domain component, and the third-level domain component

Domain Name System

- The DNS structure is like that of an inverted tree (upside-down tree);
- the root of the tree is at the top and its leaves and branches are at the bottom!
  - .Root domain
  - .et first level
  - .edu second level
  - .aau third level
- The dot that's supposed to occur after every FQDN, but it is silently assumed to be present even though it is not explicitly written

Domain Name System



Domain Name System

TLDs

- The top-level domains (TLDs) can be regarded as the first branches that we would meet on the way down from the top of our inverted tree structure.
- One can be bold and say that the top-level domains provide the categorical organization of the DNS namespace
- The TLDs can be broken down further into the generic top-level domain (e.g., .org, .com, .net, .mil, .gov, .edu, .int, .biz), country-code top-level domains (e.g., .us, .uk, .et, and .ca,

Domain Name System

Second-level domains:

- Companies, Internet service providers (ISPs), educational communities, nonprofit groups, and individuals typically acquire unique names within this level
- Examples:
  - **.telecom.et.**
  - **moe.gov.et.**
  - **aau.edu.et.**

Domain Name System

Third-level domain

- the third-level reflect hostnames or other functional uses.
  - Examples: **www.aau.edu.et.**
- • It is also common for organizations to begin the subdomain definitions from here:
  - Examples: **site.aau.edu.et.**

Domain Name System

Types of Servers

- DNS servers come in three flavors:
  - primary,
  - secondary, and
  - caching.
- Another special class of name servers consists of the so-called “root name servers.”
- Other DNS servers require the service provided by the root name servers every once in a while

Domain Name System

Primary Servers

- Primary servers are the ones considered authoritative for a particular domain.
- An authoritative server is the one on which the domain’s configuration files reside.
- When updates to the domain’s DNS tables occur, they are done on this server.
- A primary name server for a domain is simply a DNS server that knows about all hosts and subdomains existing under its domain

Domain Name System

Secondary servers

- Secondary servers work as backups and as load distributors for the primary name servers.
- Primary servers know of the existence of secondaries and send them periodic updates to the name tables.
- When a site queries a secondary name server, the secondary responds with authority.
- However, because it’s possible for a secondary to be queried before its primary can alert it to the latest changes, some people refer to secondaries as “not quite authoritative.”
- Realistically speaking, you can generally trust secondaries to have correct information.

Domain Name System

Caching servers

- Caching servers are just that: caching servers.
- They contain no configuration files for any particular domain.
- Rather, when a client host requests a caching server to resolve a name, that server will check its own local cache first.
- If it cannot find a match, it will find the primary server and ask it.
- This response is then cached

Domain Name System

BIND:

- BIND is an older and much more popular program.
- It is used on a vast majority of name-serving machines worldwide.
- BIND is currently maintained and developed by the Internet Systems Consortium (ISC).
- More can be found out about the ISC at

www.isc.org

Domain Name System

- The latest version of bind is bind9
- To install in ubuntu:

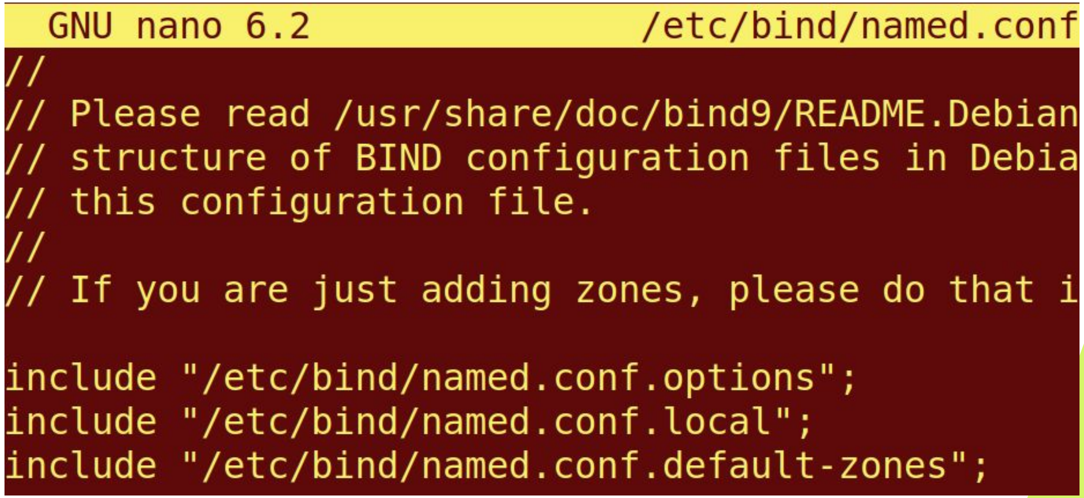
  apt-get install bind9
- Configuration files:
  - /etc/bind/named.conf
  - /etc/bind/named.conf.options
  - /etc/bind/named.conf.locals
  - /etc/bind/named.conf.default-zones
- Comments: /* */ , // , #

An address match list can include:

IP/IP with netmask/!

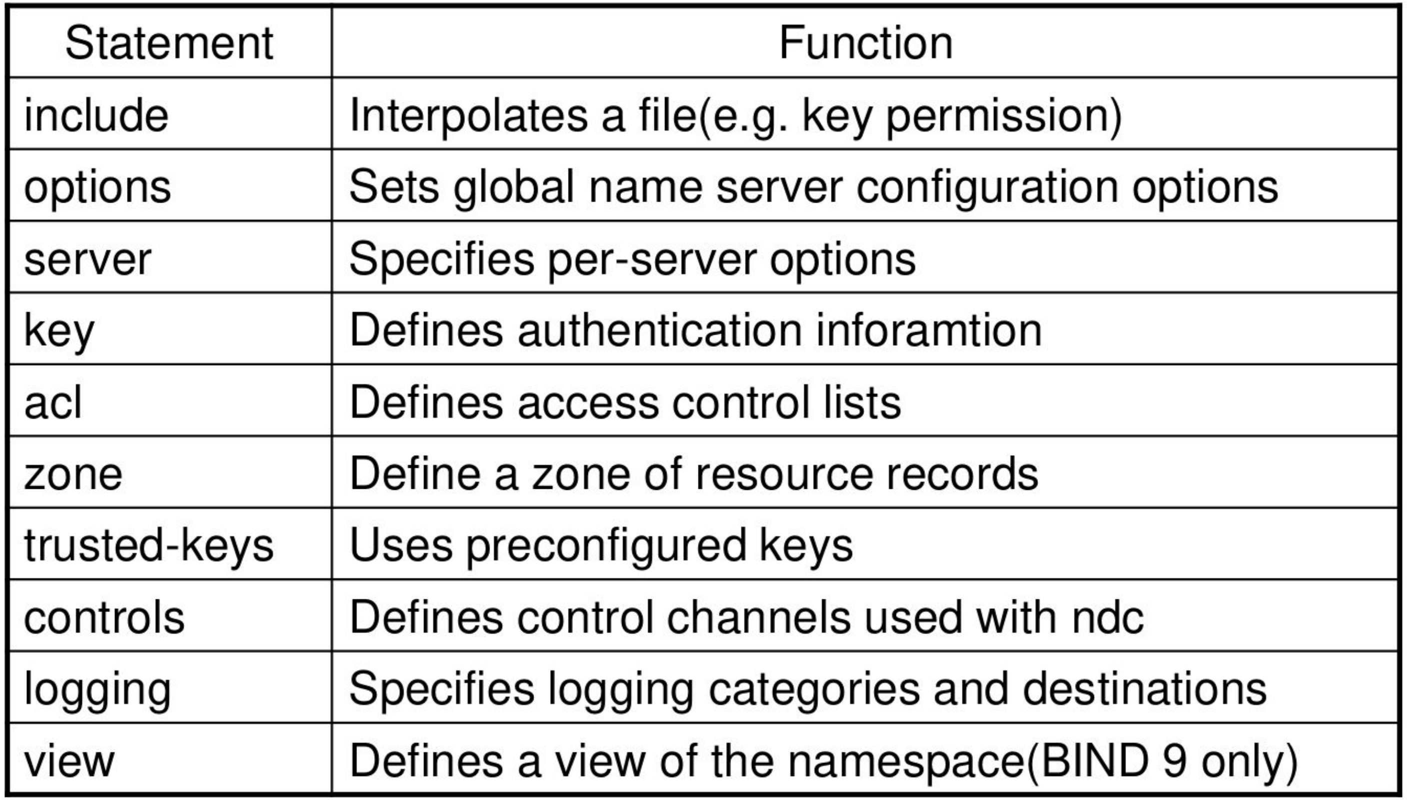
E.g. { ! 1.2.3.13; 1.2.3.24; };

{ 140.113/16; 127.0.0.1; };



```
  GNU nano 6.2                    /etc/bind/named.conf
//
// Please read /usr/share/doc/bind9/README.Debian
// structure of BIND configuration files in Debia
// this configuration file.
//
// If you are just adding zones, please do that i

include "/etc/bind/named.conf.options";
include "/etc/bind/named.conf.local";
include "/etc/bind/named.conf.default-zones";
```

Domain Name System

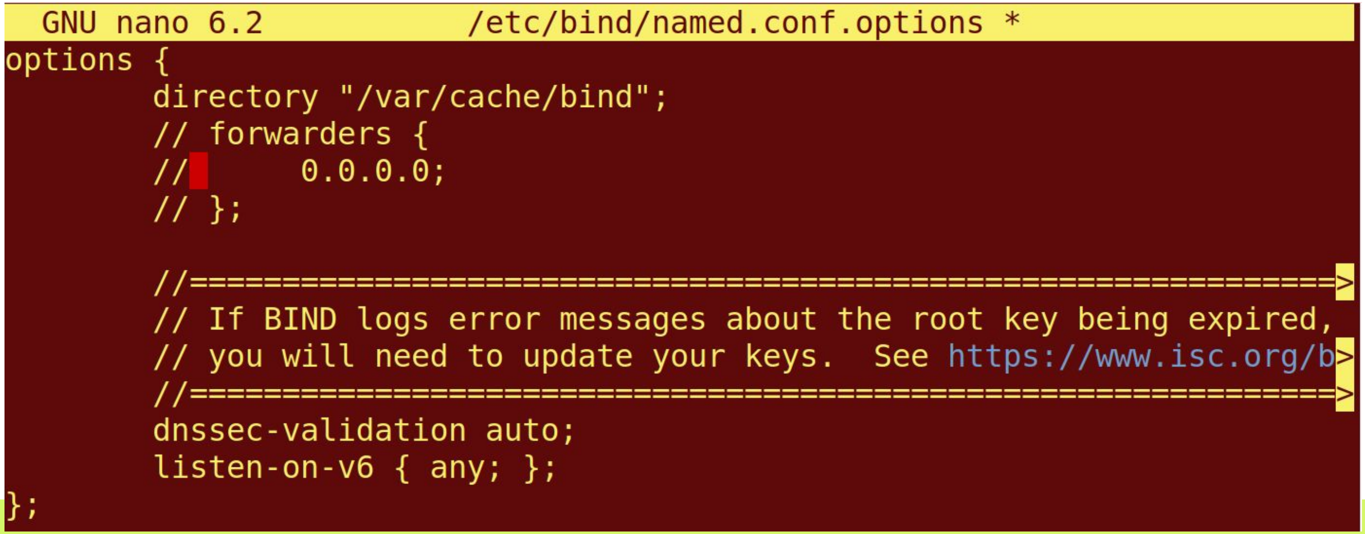


Statement	Function
include	Interpolates a file(e.g. key permission)
options	Sets global name server configuration options
server	Specifies per-server options
key	Defines authentication inforamtion
acl	Defines access control lists
zone	Define a zone of resource records
trusted-keys	Uses preconfigured keys
controls	Defines control channels used with ndc
logging	Specifies logging categories and destinations
view	Defines a view of the namespace(BIND 9 only)

Domain Name System

- options {
  option;
  option;
  …
  };

- notify yes | no; [yes]
- also-notify svrs_ips; [empty]
- recursion yes | no; [yes]
- allow-recursion { add_list }; [all hosts]



```
  GNU nano 6.2              /etc/bind/named.conf.options *
options {
        directory "/var/cache/bind";
        // forwarders {
        //      0.0.0.0;
        // };

        //=====================================================================>
        // If BIND logs error messages about the root key being expired,
        // you will need to update your keys.  See https://www.isc.org/b>
        //=====================================================================>
        dnssec-validation auto;
        listen-on-v6 { any; };
};
```

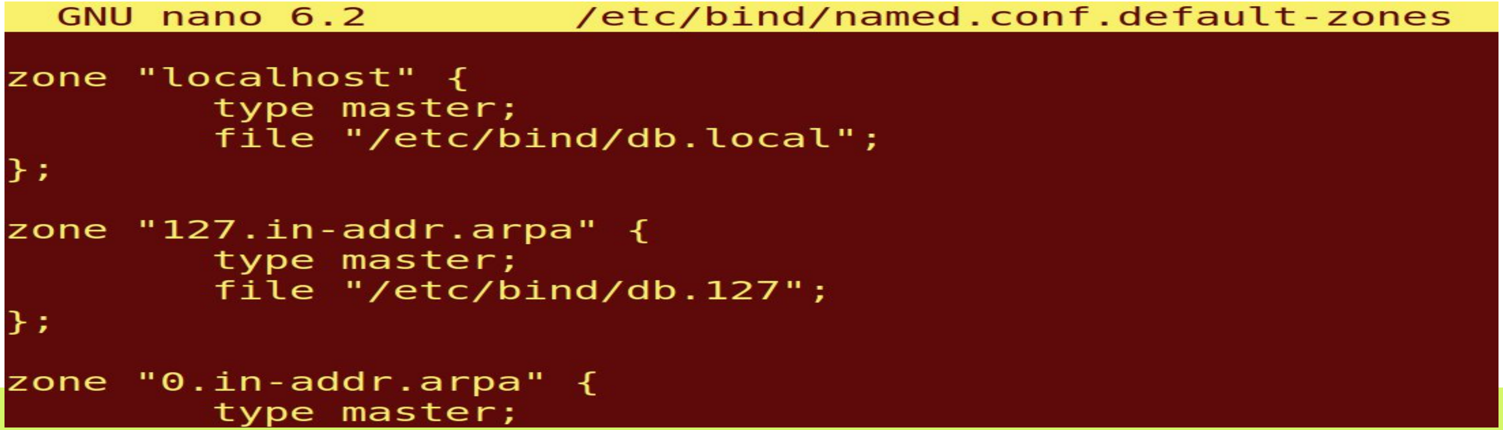
Domain Name System

- listen-on port ip_port address_match_list; [53 all]
- query-source address ip_addr port ip_port; [random]
- forwarders { in_addr; in_addr; …}; [empty]
- forward only | first; [first]
- allow-query {address_match_list; };
- allow-transfer {address_match_list;};

Domain Name System

- zone “domain_name” {
  type master|slave|stub|hint|forward;
  file “path”;
  allow-query {address_match_list; };
  allow-transfer {address_match_list; };
  allow-update {address_match_list; };
  };

- view view-name {
  match-clients {
  address_match_list };
  view_option; …
  zone_statement; …
  };



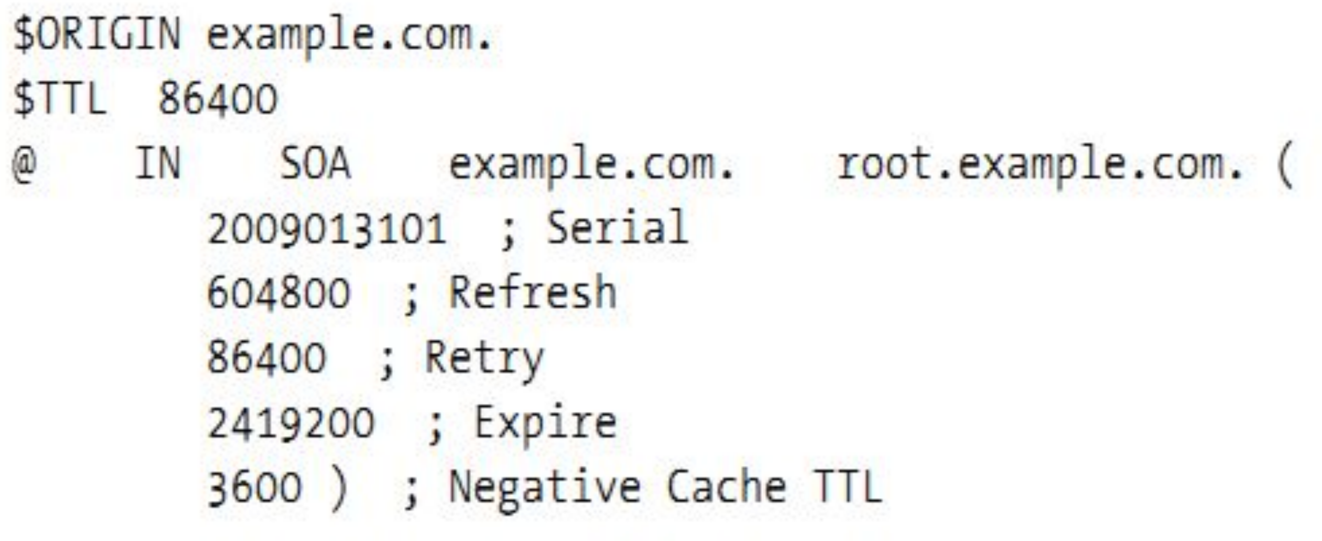
```
  GNU nano 6.2              /etc/bind/named.conf.default-zones

zone "localhost" {
        type master;
        file "/etc/bind/db.local";
};

zone "127.in-addr.arpa" {
        type master;
        file "/etc/bind/db.127";
};

zone "0.in-addr.arpa" {
        type master;
```

DNS Records

Authoritative DNS

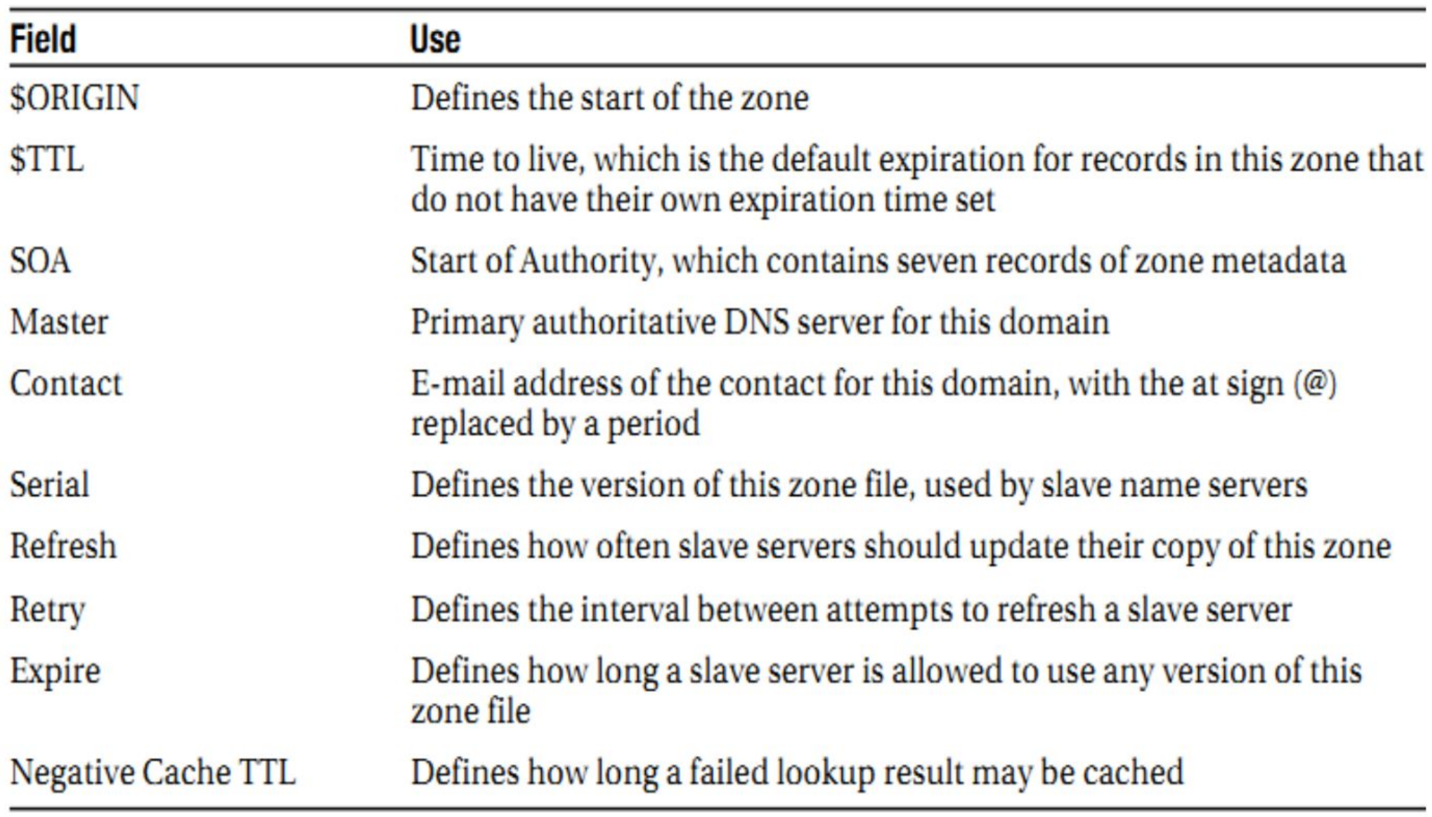
- This means the server is an authoritative source of information for a zone.
- We can use one to provide DNS resolution for our local network, on which we’ll use the example.com domain.
- We do this by defining two zones: one to provide mappings from name to address and one to provide reverse mappings, from address to name.
- For security reasons, you should not use a caching DNS server to also provide authoritative DNS servers

DNS Records



```
$ORIGIN example.com.
$TTL  86400
@    IN    SOA    example.com.    root.example.com. (
           2009013101  ; Serial
           604800   ; Refresh
           86400   ; Retry
           2419200   ; Expire
           3600 )   ; Negative Cache TTL
```

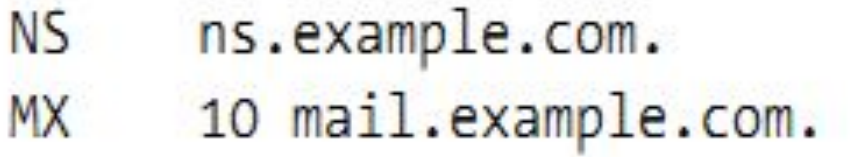
DNS Records



Field	Use
$ORIGIN	Defines the start of the zone
$TTL	Time to live, which is the default expiration for records in this zone that do not have their own expiration time set
SOA	Start of Authority, which contains seven records of zone metadata
Master	Primary authoritative DNS server for this domain
Contact	E-mail address of the contact for this domain, with the at sign (@) replaced by a period
Serial	Defines the version of this zone file, used by slave name servers
Refresh	Defines how often slave servers should update their copy of this zone
Retry	Defines the interval between attempts to refresh a slave server
Expire	Defines how long a slave server is allowed to use any version of this zone file
Negative Cache TTL	Defines how long a failed lookup result may be cached

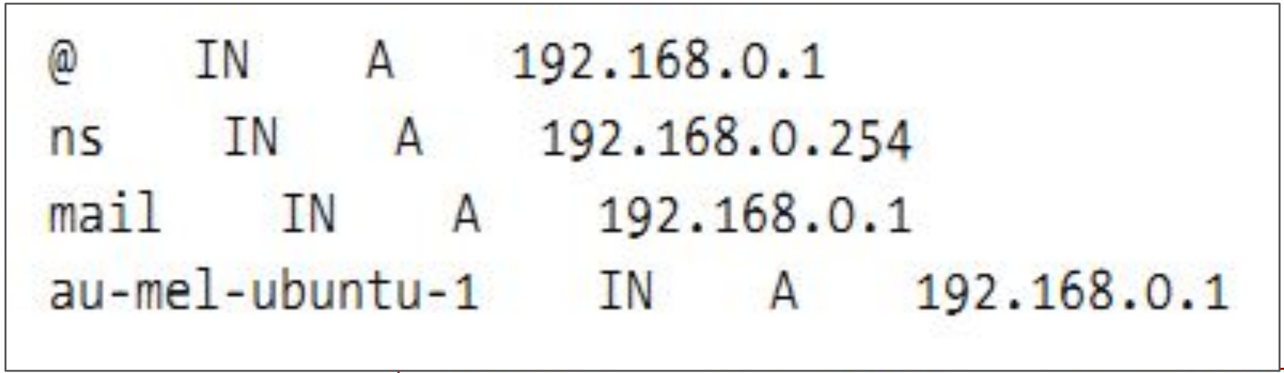
DNS Records

- You need two basic service record types to be present in your zone.
- One is the NS record, which defines which hosts act as DNS server for this domain, and the other is the MX record, which defines mail servers for this domain.
- Both records start with a blank field, as they do not define hostnames

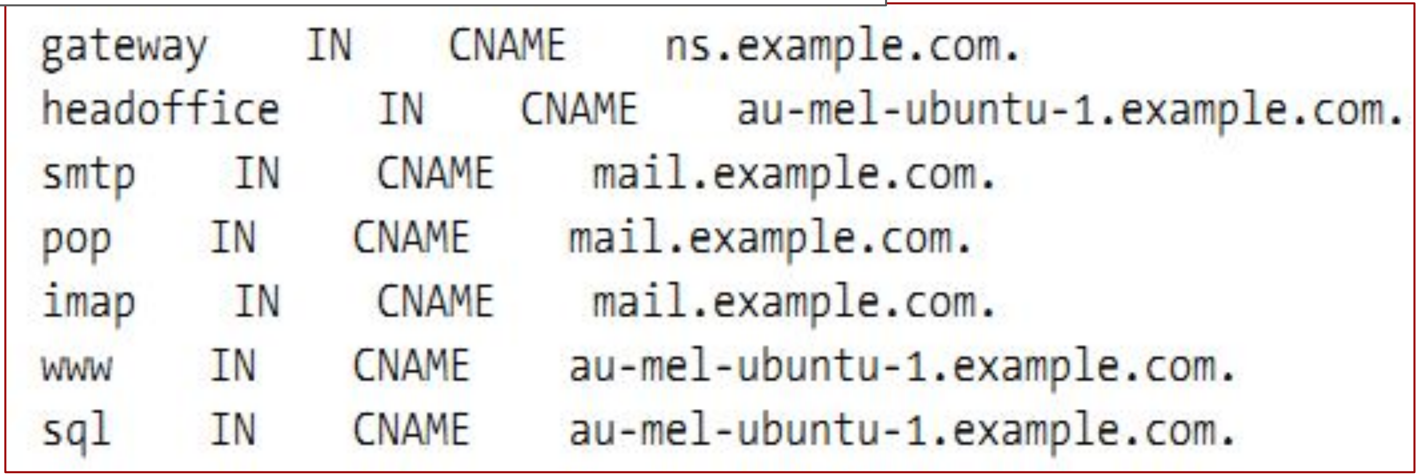


```
NS     ns.example.com.
MX     10 mail.example.com.
```

DNS Records



```
@       IN      A     192.168.0.1
ns      IN      A     192.168.0.254
mail    IN      A     192.168.0.1
au-mel-ubuntu-1    IN    A    192.168.0.1
```



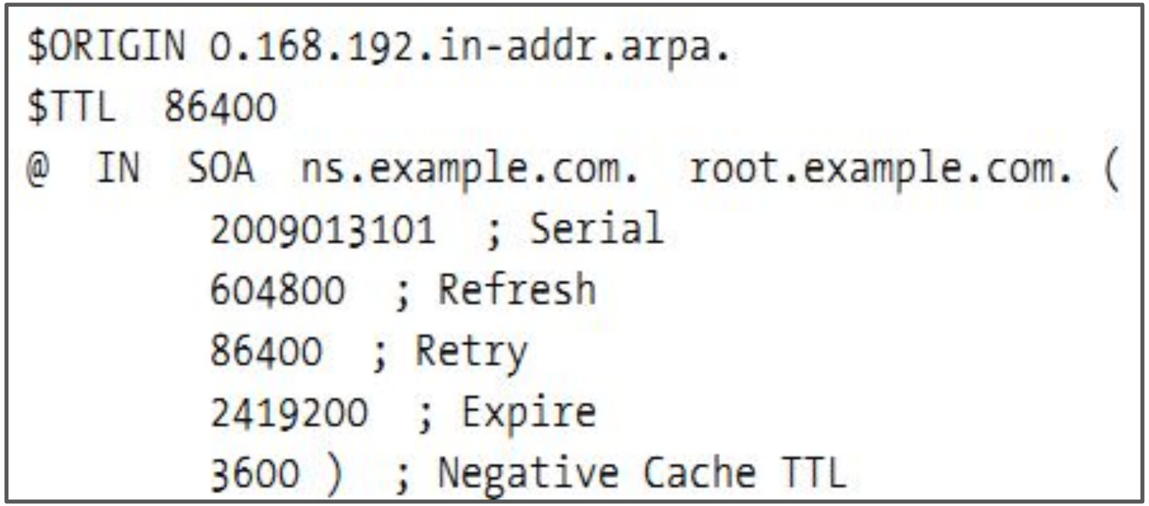
```
gateway     IN    CNAME    ns.example.com.
headoffice    IN    CNAME    au-mel-ubuntu-1.example.com.
smtp     IN    CNAME    mail.example.com.
pop     IN    CNAME    mail.example.com.
imap     IN    CNAME    mail.example.com.
www     IN    CNAME    au-mel-ubuntu-1.example.com.
sql     IN    CNAME    au-mel-ubuntu-1.example.com.
```

Reverse DNS Records

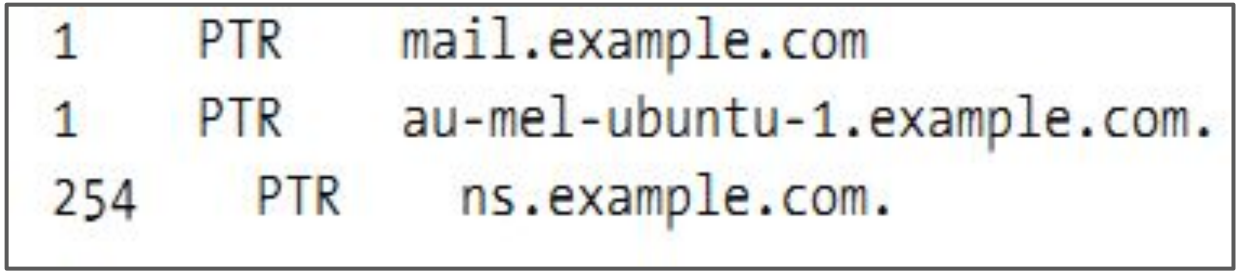
Reverse lookup zones

- IP to name resolution
- Prepend the parts of your network address to the zone, with the most significant parts to the right.
- For our network of 192.168.0.x, this results in a 0.168.192.in-addr.arpa. reverse zone name

Reverse DNS Records

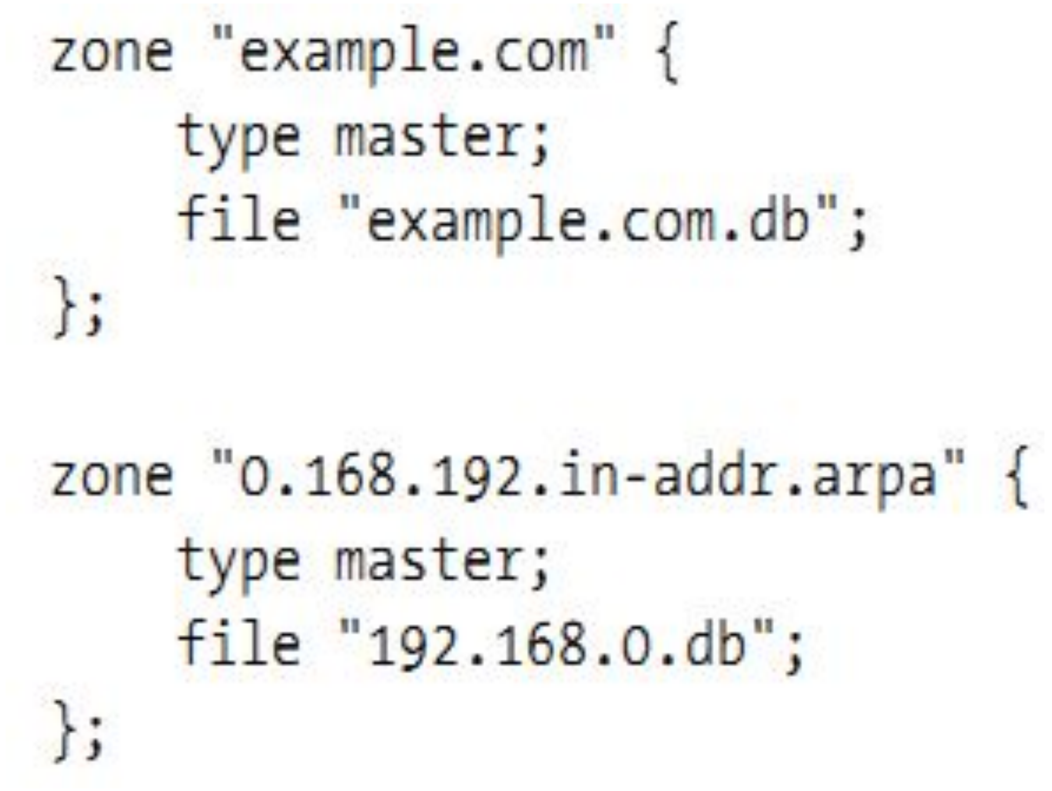


```
$ORIGIN 0.168.192.in-addr.arpa.
$TTL  86400
@  IN  SOA  ns.example.com.  root.example.com. (
        2009013101  ; Serial
        604800   ; Refresh
        86400   ; Retry
        2419200  ; Expire
        3600 )  ; Negative Cache TTL
```



```
1     PTR     mail.example.com
1     PTR     au-mel-ubuntu-1.example.com.
254     PTR     ns.example.com.
```

Reverse DNS Records



```
zone "example.com" {
    type master;
    file "example.com.db";
};

zone "0.168.192.in-addr.arpa" {
    type master;
    file "192.168.0.db";
};
```

Tools to test

- Dig example.com
- Nslookup example.com

READING ASSIGNMENT

- Configuring slave DNS server
- Configuring cache DNS server
- Stealth DNS server
- Forward DNS server

end.