Web Server Administration

Chapter 22

Randy Connolly and Ricardo Hoar

Fundamentals of Web Development

Objectives

1 Web Server Hosting Options

2 Domain and Name Server Administration

3 Linux and Apache Configuration

4 Apache Request/Response

5 Web Monitoring and Analytics

Section 1 of 5

WEB SERVER HOSTING OPTIONS

Hosting

Development vs Production

Since you have been working with PHP, you have already worked with some sort of web server.

However, most server tools that simplify matters for development purposes (like WAMP) gloss over the nitty-gritty details of an Apache server.

In a real-world scenario, you must be aware of advanced configuration options, ideas, and tools that ensure your server is deployed and maintained according to established best practices.

Types of Hosting

3 categories

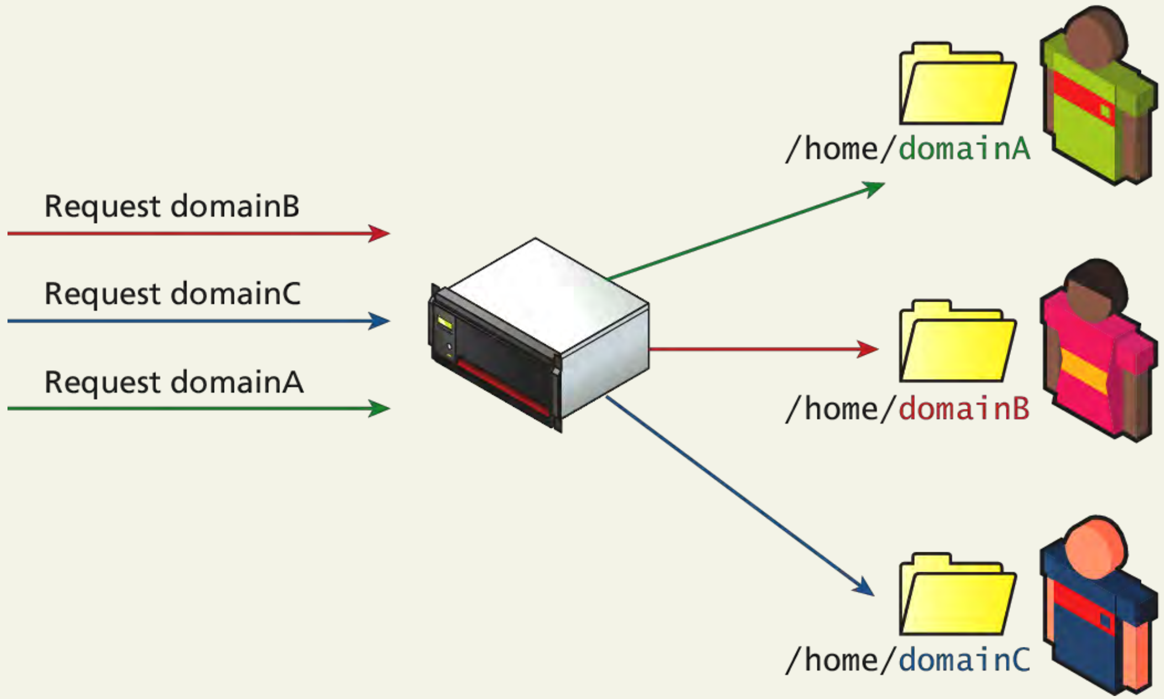
The three broad categories of web hosting are:

- Shared Hosting
- Collocated Hosting
- Dedicated Hosting

Shared Hosting

Cost effective Hosting

Shared hosting is renting space for your site on a server that will host many sites on the same machine



Shared Hosting

Sharing is ok

Shared hosting is normally the least expensive, least functional, and most common type of hosting solution, especially for small websites.

This class of hosting is divided into two categories:

- simple shared hosting and
- virtualized shared hosting.

Simple Shared Hosting

The Cheapest

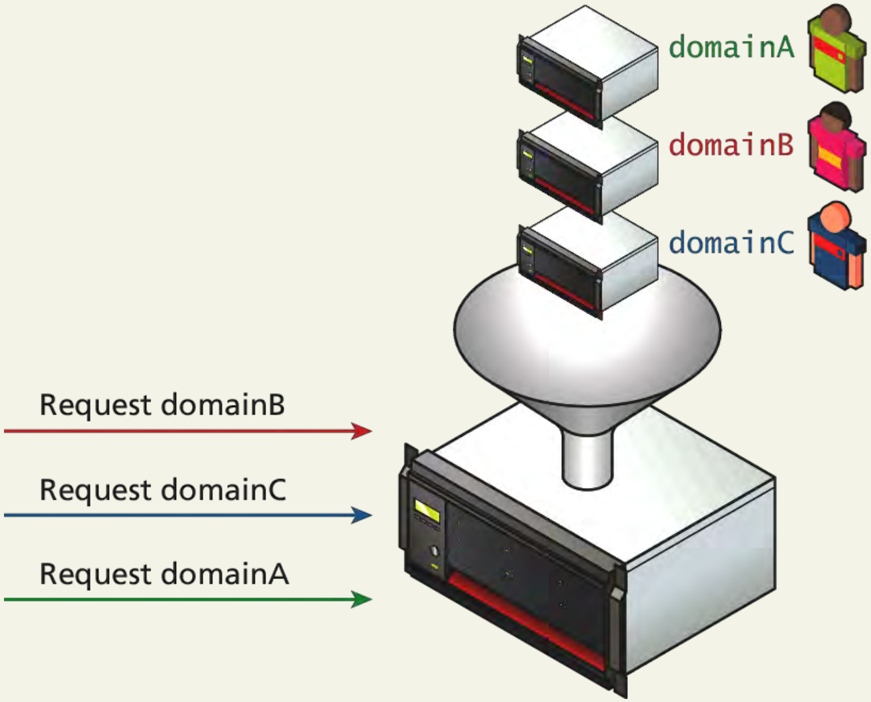
Simple shared hosting is a hosting environment in which clients receive access to a folder on a web server, but cannot increase their privileges to configure any part of the operating system, web server, or database.

The disadvantages of simple shared hosting are many. Lack of control, poor performance, and security threats make shared hosting a bad idea for a serious website.

Virtualized Shared Hosting

Better, but still cost effective

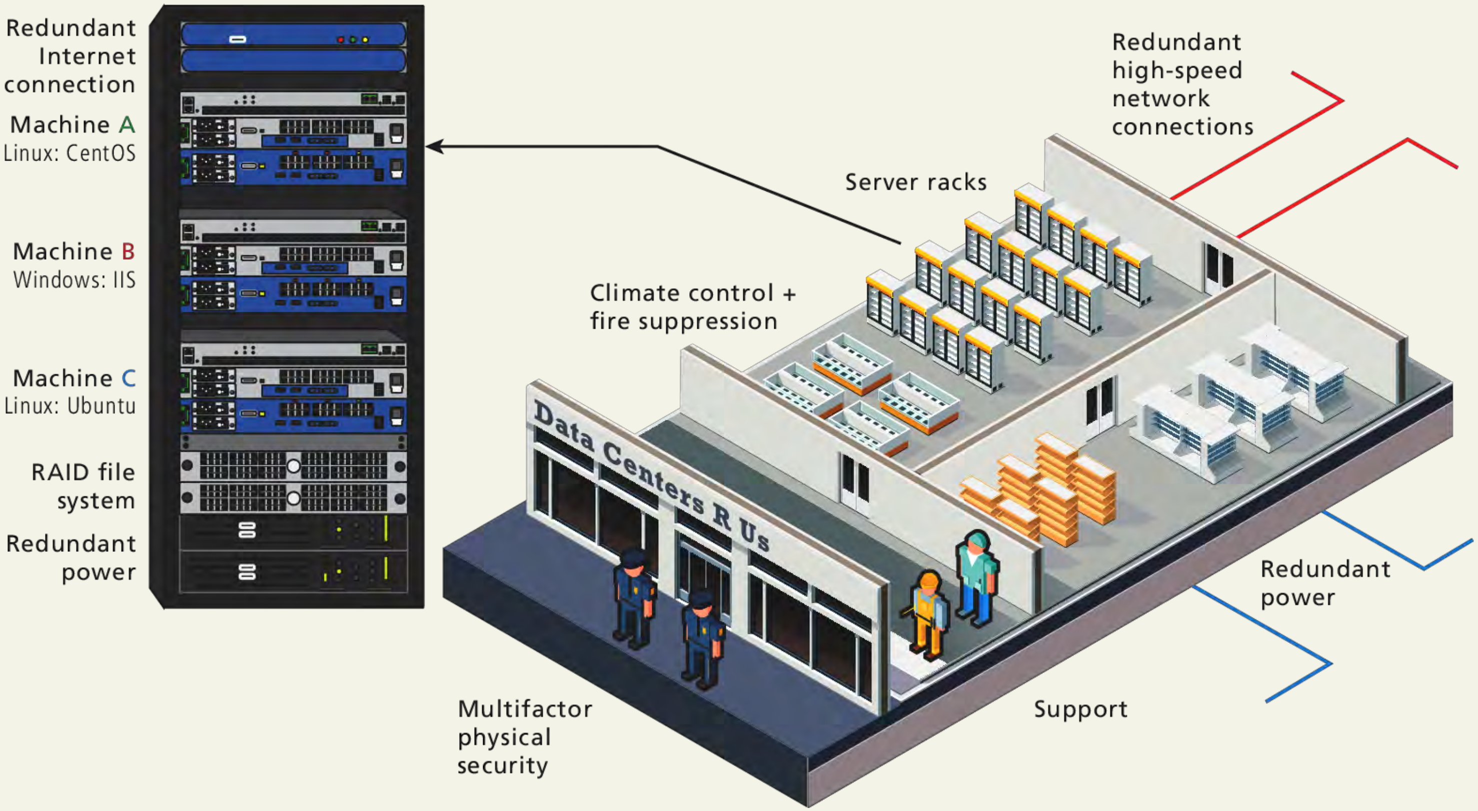
Virtualized shared hosting is a variation on the shared hosting scheme, where instead of being given a username and a home directory on a shared server, you are given a virtual server, with root access



Dedicated Hosting

Almost your machine

Dedicated hosting is when a server is rented to you in its entirety inside the data center



Dedicated Hosting

Almost your machine

Data centers are normally located to take advantage of nearby Internet Exchange Points and benefit from redundant connections.

You are given a complete physical machine to control, removing the possible inequity that can arise when you share the CPU and RAM with other users.

The disadvantage of dedicated hosting is the lack of control over the hardware, and a restriction on accessing the hardware.

Collocated Hosting

Touch the machine

Collocated hosting is almost like dedicated hosting, except rather than rent a machine, you outright build, own, and manage the machine yourself.

The advantage of collocated hosting goes beyond a dedicated server with not only full control over the OS, software version, firewalls, and policies but also the physical machine.

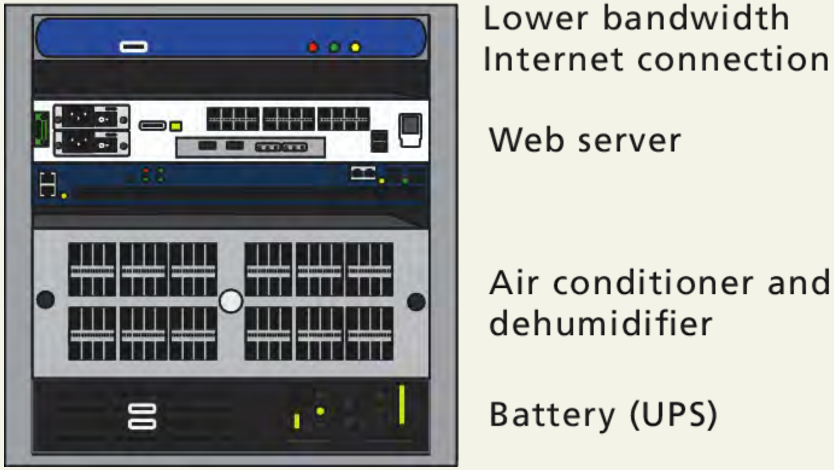
The disadvantage of collocated systems is that you must control everything yourself, with little to no support from a third party and they are costly

In House Hosting

Do everything yourself

Many companies do use a low-cost, in-house hosting environment for development, preproduction, and sandbox environments.

In practice, though, many small companies' in-house data centers are just closets with an air conditioner, unsecured, and without any redundancies.


Lower bandwidth Internet connection
Web server
Air conditioner and dehumidifier
Battery (UPS)

Cloud Hosting

Ignore the man behind the curtain

Cloud hosting is the newest buzzword in shared hosting services.

The advantages are

- scalability, where more computing and data storage are needed and
- The redundancy of a distributed solution

Unfortunately, many providers are cashing in on the latest buzzwords without the benefits.

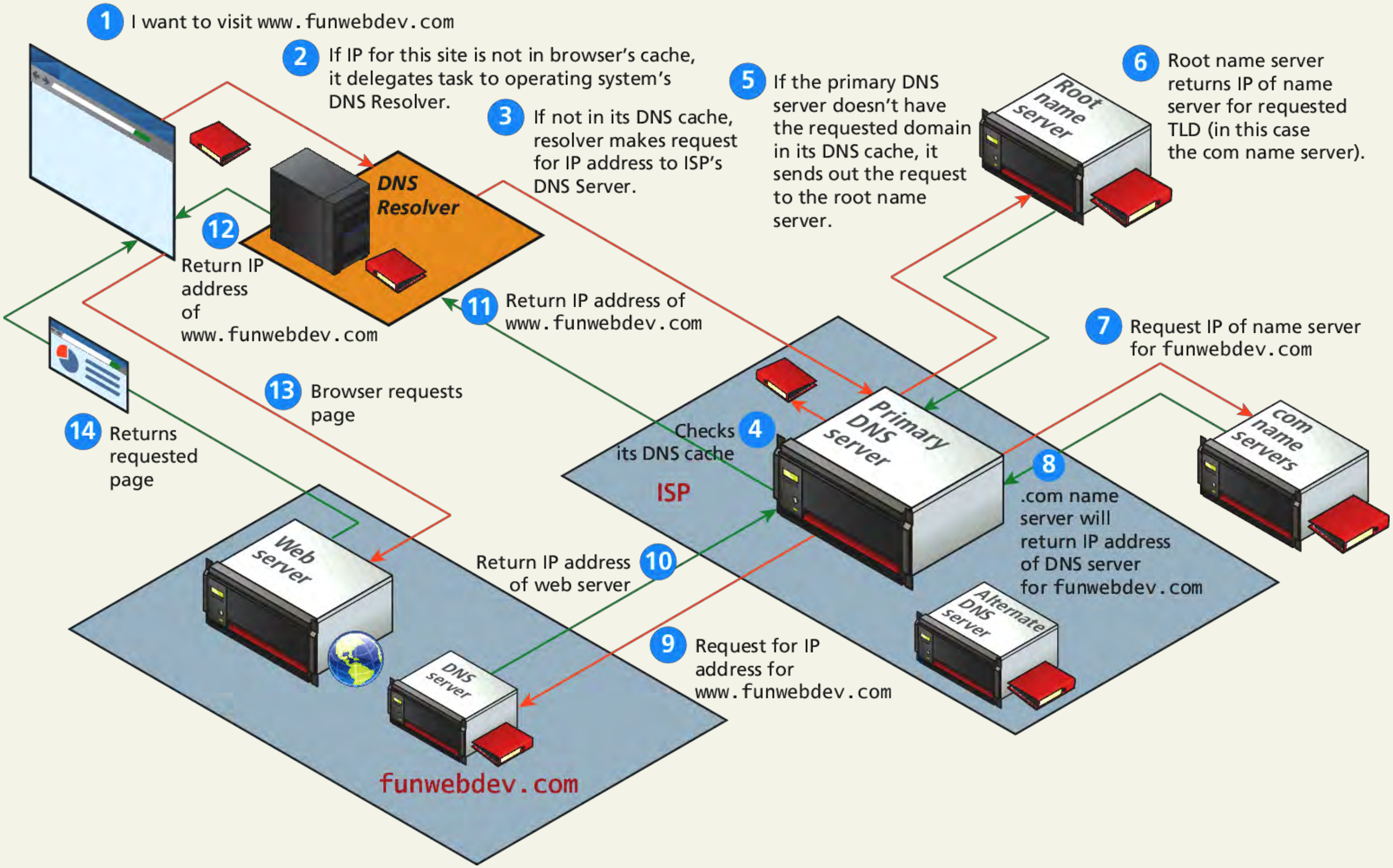
At the end of the day a request for your website has to be answered by a physical machine with access to RAM, file system, and an OS.

Section 2 of 5

DOMAIN AND NAME SERVER ADMINISTRATION

Domain Name System

Better than remembering IP addresses



1 I want to visit www.funwebdev.com

2 If IP for this site is not in browser's cache, it delegates task to operating system's DNS Resolver.

3 If not in its DNS cache, resolver makes request for IP address to ISP's DNS Server.

4 Checks its DNS cache

5 If the primary DNS server doesn't have the requested domain in its DNS cache, it sends out the request to the root name server.

6 Root name server returns IP of name server for requested TLD (in this case the com name server).

7 Request IP of name server for funwebdev.com

8 .com name server will return IP address of DNS server for funwebdev.com

9 Request for IP address for www.funwebdev.com

10 Return IP address of web server

11 Return IP address of www.funwebdev.com

12 Return IP address of www.funwebdev.com

13 Browser requests page

14 Returns requested page

Registering a Domain Name

Step one to your fortune

You only lease the right to use the name exclusively for a period, and must renew periodically.

Registrars are companies that register domain names, on your behalf (the registrant), under the oversight of ICANN.

Some popular registrars include GoDaddy, TuCows, and Network Solutions, where you can expect to pay from $10.00 per year per domain name.

Registering a Domain Name

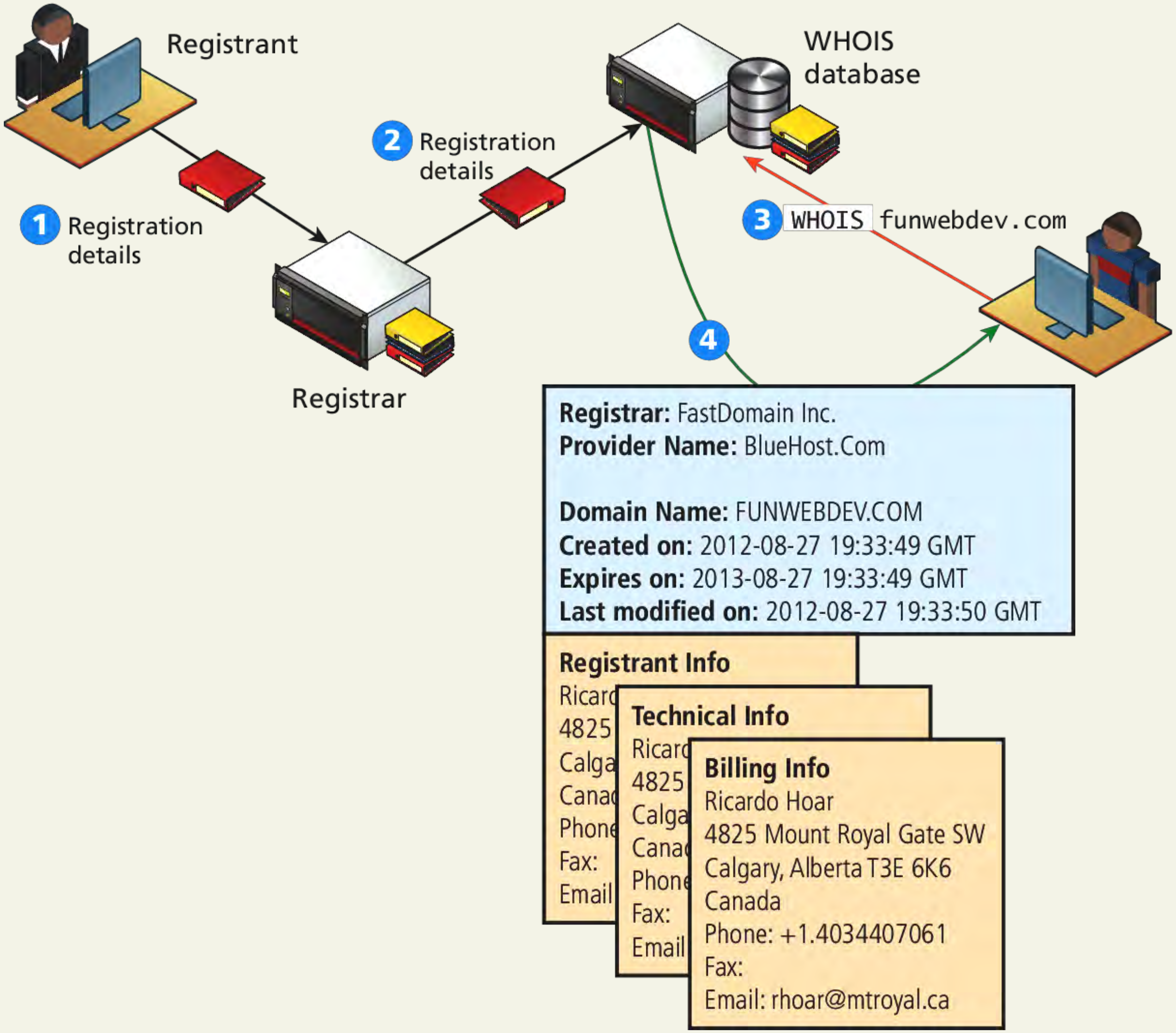
WHOIS

The registrars must collect and maintain your information in a database of WHOIS records that includes three levels of contact (registrant, technical, and billing), who are often the same person.

Anyone can try and find out who owns a domain by running the WHOIS command and reading the output.

Whois

A Visualization



Registrar: FastDomain Inc.
Provider Name: BlueHost.Com

Domain Name: FUNWEBDEV.COM
Created on: 2012-08-27 19:33:49 GMT
Expires on: 2013-08-27 19:33:49 GMT
Last modified on: 2012-08-27 19:33:50 GMT

Registrant Info
Ricarc
4825
Calga
Canac
Phone
Fax:
Email

Technical Info
Ricarc
4825
Calga
Canac
Phone
Fax:
Email

Billing Info
Ricardo Hoar
4825 Mount Royal Gate SW
Calgary, Alberta T3E 6K6
Canada
Phone: +1.4034407061
Fax:
Email: rhoar@mtroyal.ca

Whois

Private Visualization

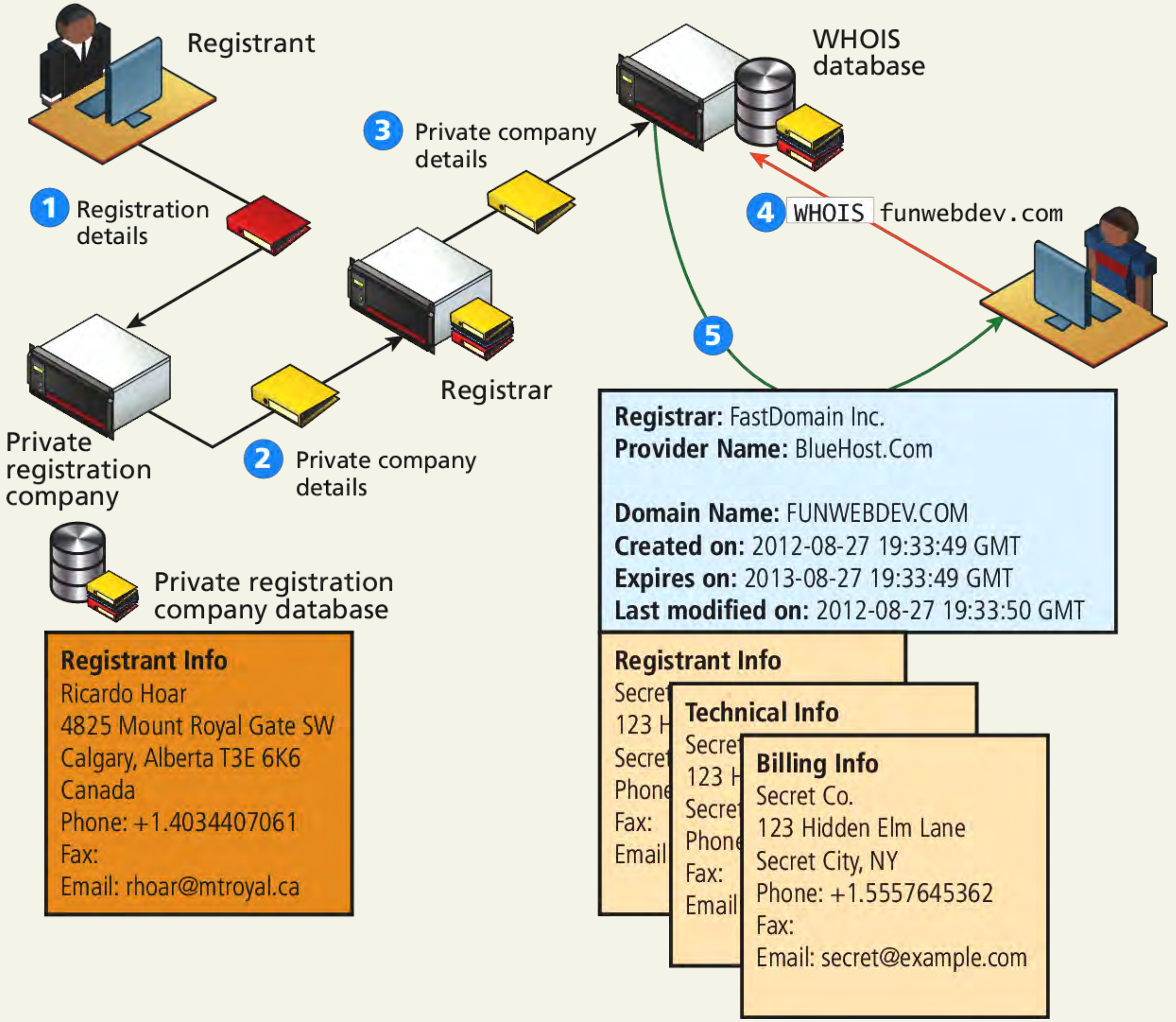
Many registrars provide **private registration** services, which broker a deal with a private company as an intermediary to register the domain on your behalf.

The private registration company keeps your real contact information on their own servers because they must know who to contact if the need arises.

These private registrants will turn your information over to authorities upon request

Whois

Private Visualization



Registrant

1 Registration details

Private registration company

2 Private company details

Registrar

3 Private company details

WHOIS database

4 `WHOIS funwebdev.com`

5

Private registration company database

Registrant Info
Ricardo Hoar
4825 Mount Royal Gate SW
Calgary, Alberta T3E 6K6
Canada
Phone: +1.4034407061
Fax:
Email: rhoar@mtroyal.ca

Registrar: FastDomain Inc.
Provider Name: BlueHost.Com

Domain Name: FUNWEBDEV.COM
Created on: 2012-08-27 19:33:49 GMT
Expires on: 2013-08-27 19:33:49 GMT
Last modified on: 2012-08-27 19:33:50 GMT

Registrant Info
Secret
123 H
Secret
Phone
Fax:
Email

Technical Info
Secret
123 H
Secret
Phone
Fax:
Email

Billing Info
Secret Co.
123 Hidden Elm Lane
Secret City, NY
Phone: +1.5557645362
Fax:
Email: secret@example.com

Updating the Name Servers

Easy to use, a little tricky to update

The single most important thing you do with your registrar is control the name servers associated with the domain name.

Your web host will provide name servers which then have to get registered with the registrar you used when you leased the domain.

When you update your name server, the registrar, on your behalf, updates your name server records on the top-level domain (TLD) name servers

Checking Name Servers

Some little tricks

Updating records in DNS may require at least 48 hours to ensure that the changes have propagated throughout the system.

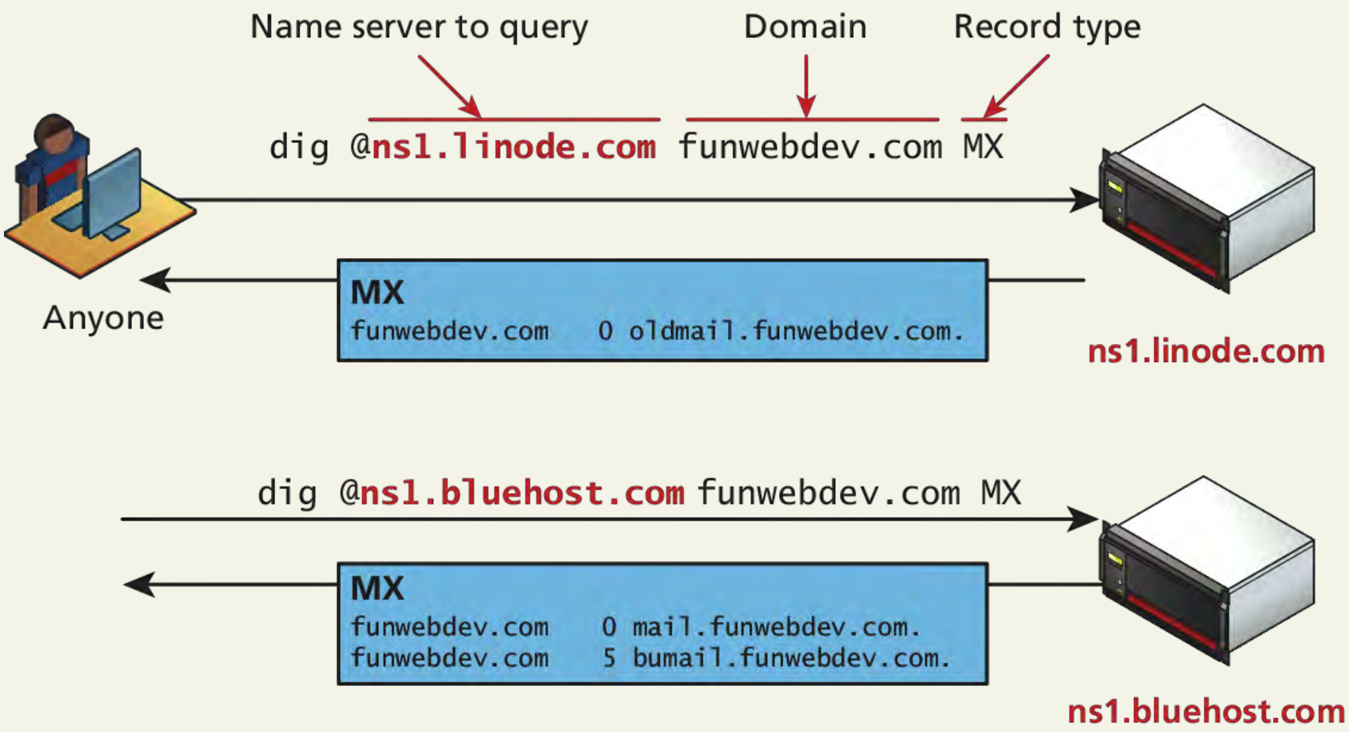
After updating your name servers with the registrar, it's a good practice to "dig" on your TLD servers to confirm that the changes have been made.

Dig is a command that lets you ask a particular name server about records of a particular type for any domain.

Checking Name Servers

Dig it

```
dig @ns1.linode.com funwebdev.com MX
```



```
dig @ns1.linode.com funwebdev.com MX
```

```
MX
funwebdev.com    0 oldmail.funwebdev.com.
```

```
dig @ns1.bluehost.com funwebdev.com MX
```

```
MX
funwebdev.com    0 mail.funwebdev.com.
funwebdev.com    5 bumail.funwebdev.com.
```

DNS Record Types

Host, Mail Server, Name Server, Alias, …

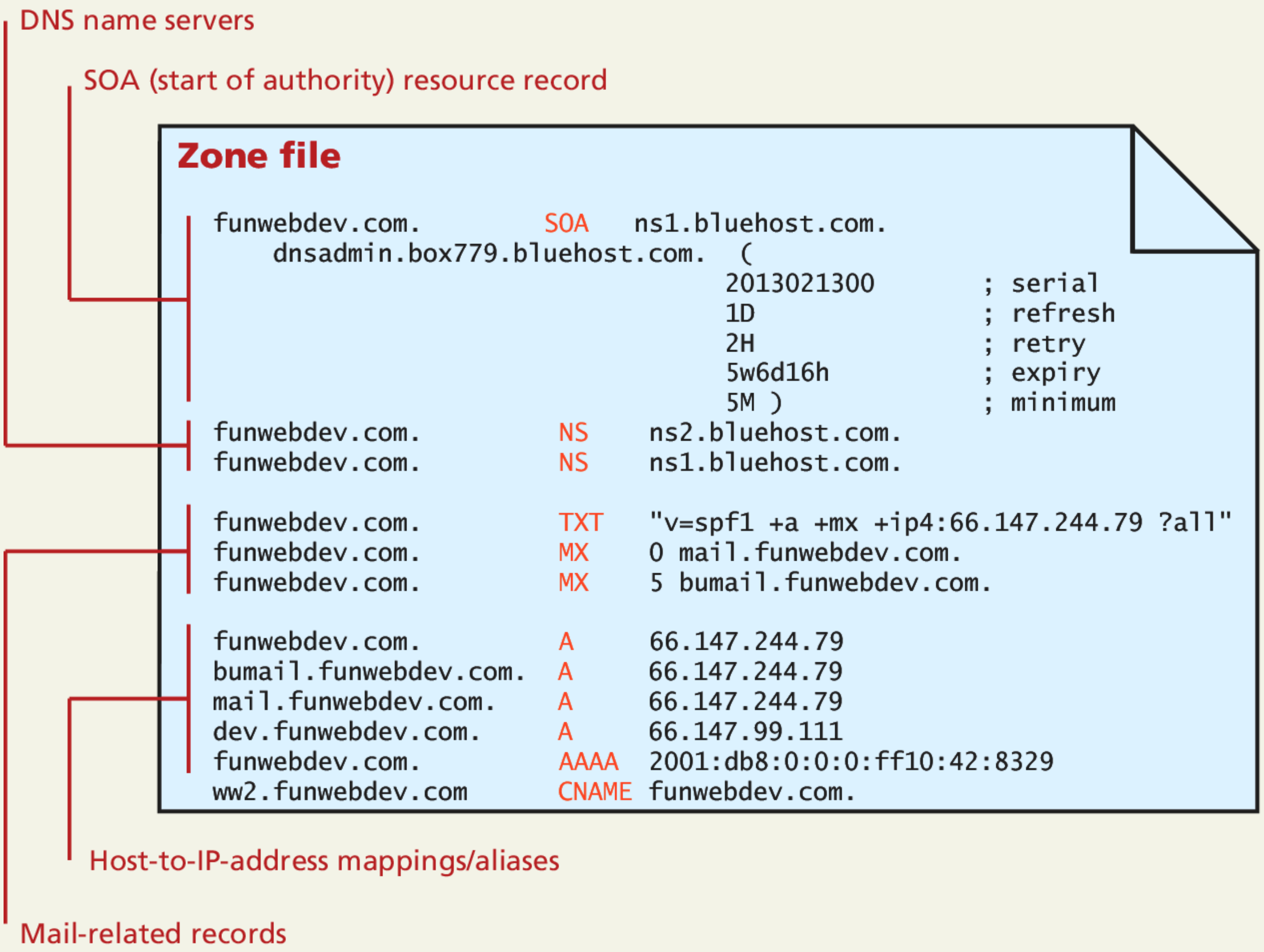
In practice, all of a domain's records are stored in a single file called the DNS **zone file**.

There are six primary types of records

- **A/AAA,**
- **CName,**
- **MX,**
- **NS,**
- **SOA,** and
- **TXT/SPF**

DNS Record Types

Zone file



DNS name servers

SOA (start of authority) resource record

Zone file

```
funwebdev.com.         SOA   ns1.bluehost.com.
    dnsadmin.box779.bluehost.com.  (
                                   2013021300        ; serial
                                   1D                ; refresh
                                   2H                ; retry
                                   5w6d16h           ; expiry
                                   5M )              ; minimum
funwebdev.com.          NS    ns2.bluehost.com.
funwebdev.com.          NS    ns1.bluehost.com.

funwebdev.com.          TXT   "v=spf1 +a +mx +ip4:66.147.244.79 ?all"
funwebdev.com.          MX    0 mail.funwebdev.com.
funwebdev.com.          MX    5 bumail.funwebdev.com.

funwebdev.com.          A     66.147.244.79
bumail.funwebdev.com.   A     66.147.244.79
mail.funwebdev.com.     A     66.147.244.79
dev.funwebdev.com.      A     66.147.99.111
funwebdev.com.          AAAA  2001:db8:0:0:0:ff10:42:8329
ww2.funwebdev.com       CNAME funwebdev.com.
```

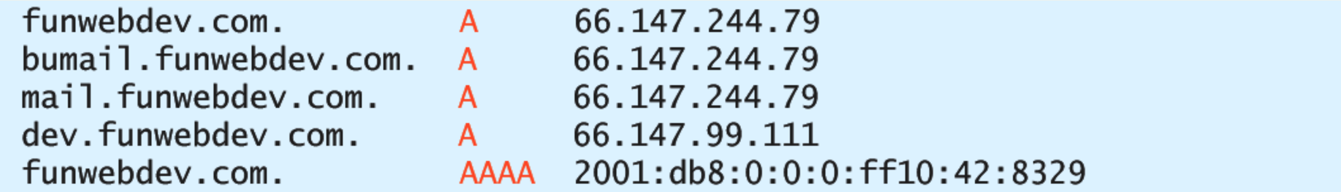
Host-to-IP-address mappings/aliases

Mail-related records

DNS Record Types

A and AAAA Records

A records and **AAAA records** are identical except *A* records use IPv4 addresses and *AAAA* records use IPv6.



```
funwebdev.com.          A      66.147.244.79
bumail.funwebdev.com.   A      66.147.244.79
mail.funwebdev.com.     A      66.147.244.79
dev.funwebdev.com.      A      66.147.99.111
funwebdev.com.          AAAA   2001:db8:0:0:0:ff10:42:8329
```

Both of them simply associate a hostname with an IP address.

These are the most common queries, performed whenever a user requests a domain through a browser.

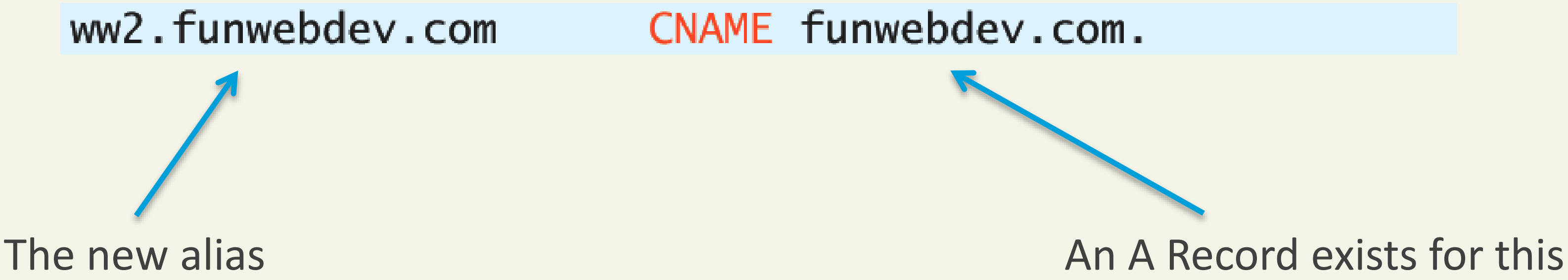
DNS Record Types

CNAME Records

Canonical Name (CName) records is allow you to point multiple subdomains to an existing *A* record.

This allows you to update all your domains at once by changing the one *A* record. However, it doubles the number of queries required to get resolution for your domain, making *A* records the preferred technique.

It is sometimes called an alias.



```
ww2.funwebdev.com      CNAME funwebdev.com.
```

The new alias

An A Record exists for this

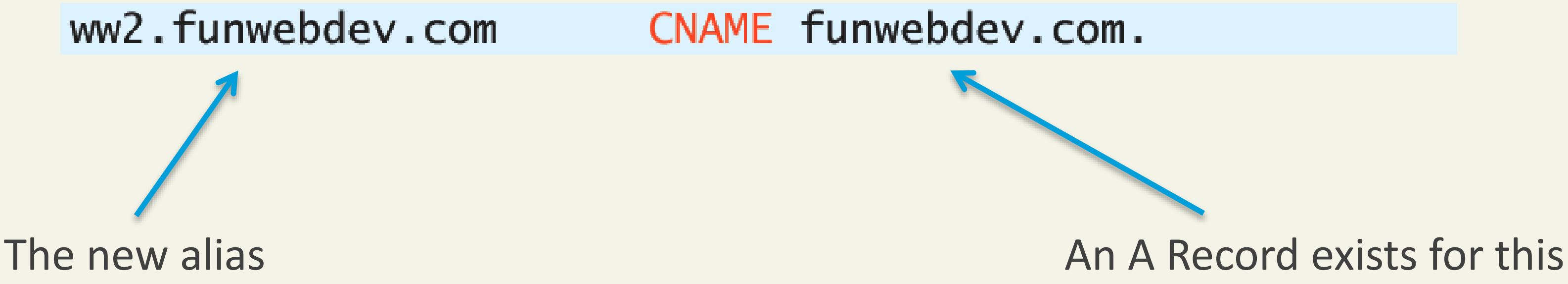
DNS Record Types

CNAME Records

Canonical Name (CName) records is allow you to point multiple subdomains to an existing *A* record.

This allows you to update all your domains at once by changing the one *A* record. However, it doubles the number of queries required to get resolution

It is sometimes called an alias.



```
ww2.funwebdev.com       CNAME funwebdev.com.
```

The new alias

An A Record exists for this

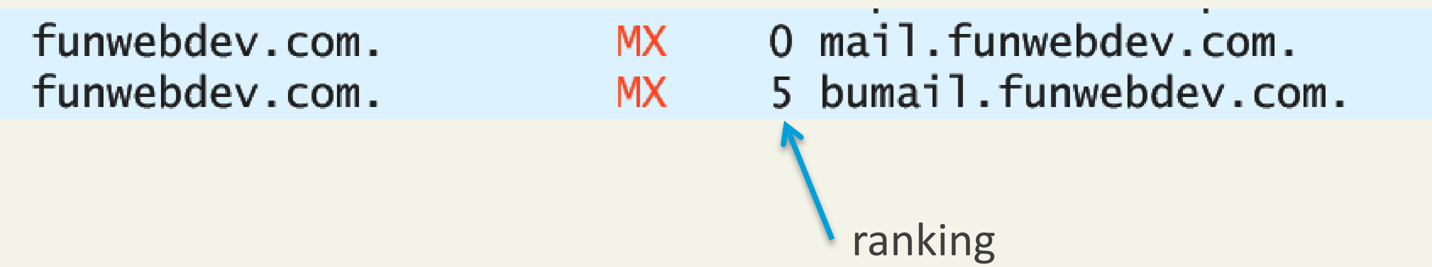
DNS Record Types

Mail Records

Mail Exchange (MX) records are the records that provide the location of the Simple Mail Transfer Protocol (SMTP) servers to receive email for this domain.

SMTP allows redundant mail servers for load distribution or backup purposes. To support that feature, MX records not only require an IP address but also a ranking.

When trying to deliver mail, the lowest numbered servers are tried first, and only if they are down, will the higher ones be used.



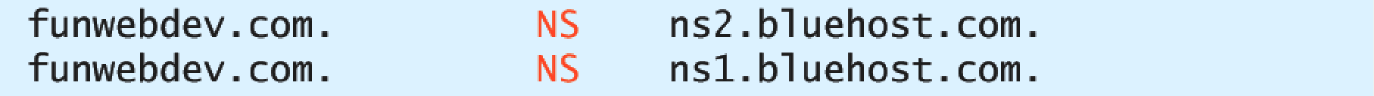
```
funwebdev.com.          MX     0 mail.funwebdev.com.
funwebdev.com.          MX     5 bumail.funwebdev.com.
```

ranking

DNS Record Types

Authoritative Records

Name server (NS) records are the essential records that tell everyone what name servers to use for this domain. There can be (and should be) multiple name servers listed for redundancy.

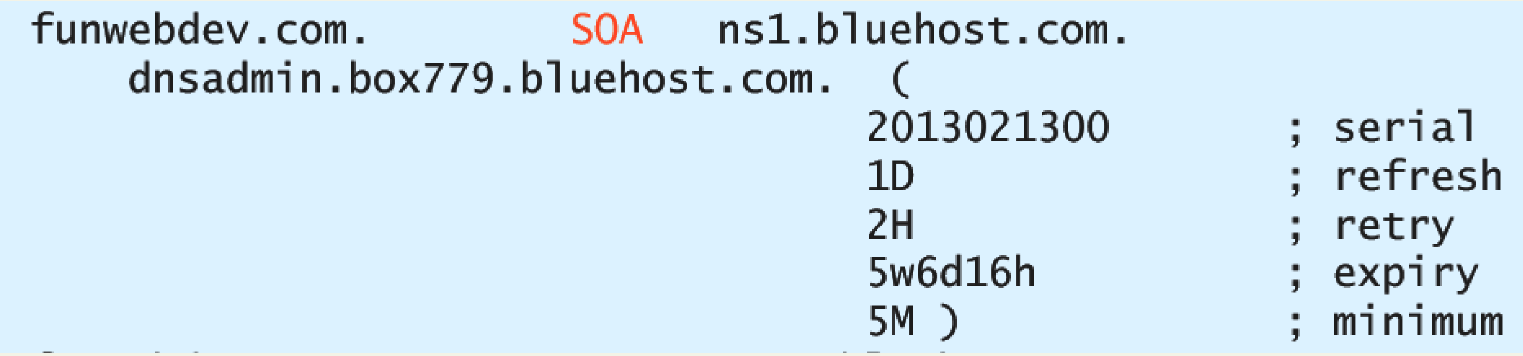


```
funwebdev.com.          NS    ns2.bluehost.com.
funwebdev.com.          NS    ns1.bluehost.com.
```

Start of Authority (SOA) record contains information about how long this record is valid [called time to live (TTL)], together with a serial number that gets incremented with each update to help synchronize DNS

DNS Record Types

Start of Authority (SOA) record



```
funwebdev.com.         SOA   ns1.bluehost.com.
    dnsadmin.box779.bluehost.com.  (
                                     2013021300       ; serial
                                     1D               ; refresh
                                     2H               ; retry
                                     5w6d16h          ; expiry
                                     5M )             ; minimum
```

DNS Record Types

Validation Records

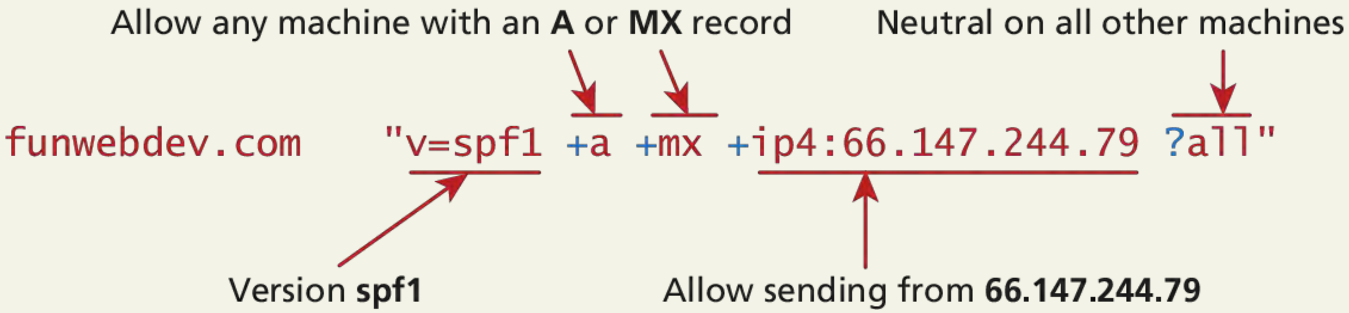
TXT and **Sender Policy Framework (SPF) records** are used to reduce email spam by providing another mechanism to validate your mail servers for the domain.

SPF records appear as both SPF and TXT records.

The value is a string, enclosed in double quotes (" ") that starts with **v=spf1** (the version) and uses space-separated selectors with modifiers to define which machines should be allowed to send email as this domain.

DNS Record Types

Validation Records



Allow any machine with an **A** or **MX** record

Neutral on all other machines

```
funwebdev.com     "v=spf1 +a +mx +ip4:66.147.244.79 ?all"
```

Version **spf1**

Allow sending from **66.147.244.79**

Reverse DNS

in-addr.apra

Reverse DNS is the reverse process, whereby you get a domain name from an IP address

A **pointer (PTR) record** is created with the IP address prepended in reverse order to the domain **in-addr.arpa**

66.147.244.79 becomes the PTR entry

funwebdev.com PTR 79.244.147.66.**in-addr.apra**

Now, when a mail server wants to determine if a received email is spam or not, they recreate the **in-addr.apra** hostname from the IP in the email and resolve it like any other DNS request based on the domain it claims to be from.

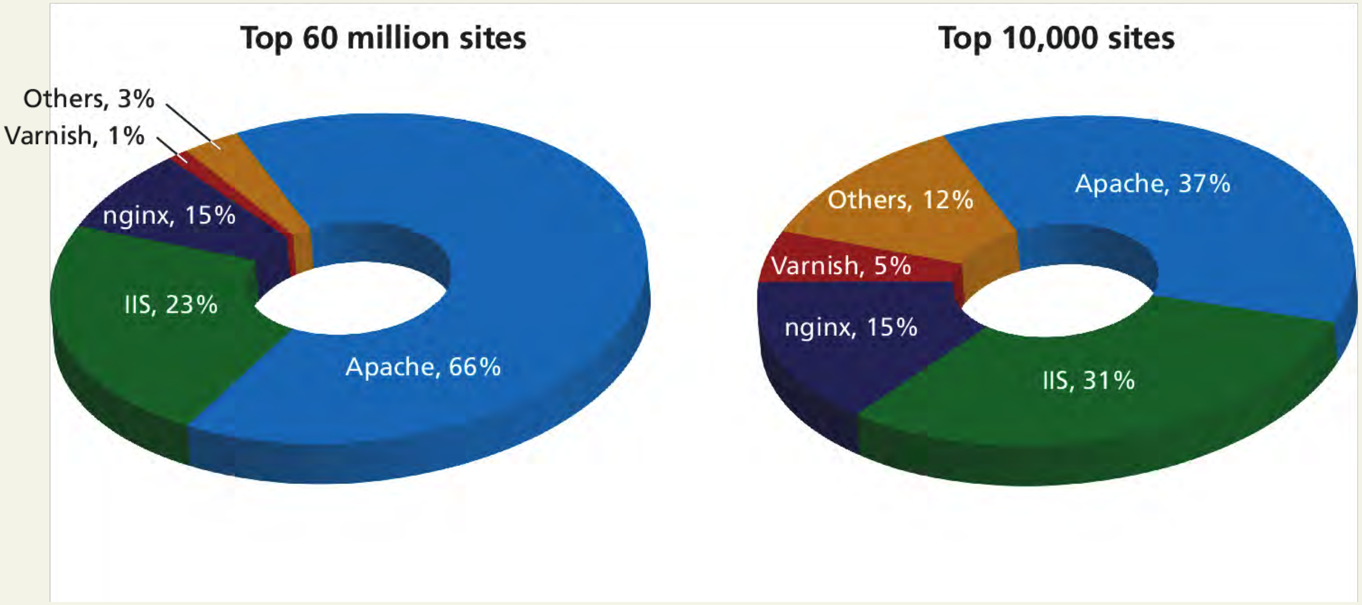
Section 3 of 5

LINUX AND APACHE CONFIGURATION

Apache

The world's most popular webserver

Web server software like Apache is responsible for handling HTTP requests on your server.



Apache

Configuration

Apache can be configured through two key locations

- When Apache is started or restarted, it parses the **root configuration file**, which is normally writable by only root users (stored in **/etc/httpd.conf**, or somewhere similar).
- **directory-level configuration files** are permitted which can change the behavior of the server without having to restart Apache. The files are normally named **.htaccess** (hypertext access), and they can reside inside any of the public folders served by Apache.

Daemons

Apache runs all the time

A **daemon** is software that runs forever in the background of an operating system and normally provides one simple **service**. Daemons on Linux include sshd, httpd, mysqld, as well as many others.

To start, stop and restart the Apache daemon from the command line in Linux, the root user can enter these commands:

```
/etc/init.d/httpd start
```

```
/etc/init.d/httpd stop
```

```
/etc/init.d/httpd restart
```

Managing Daemons

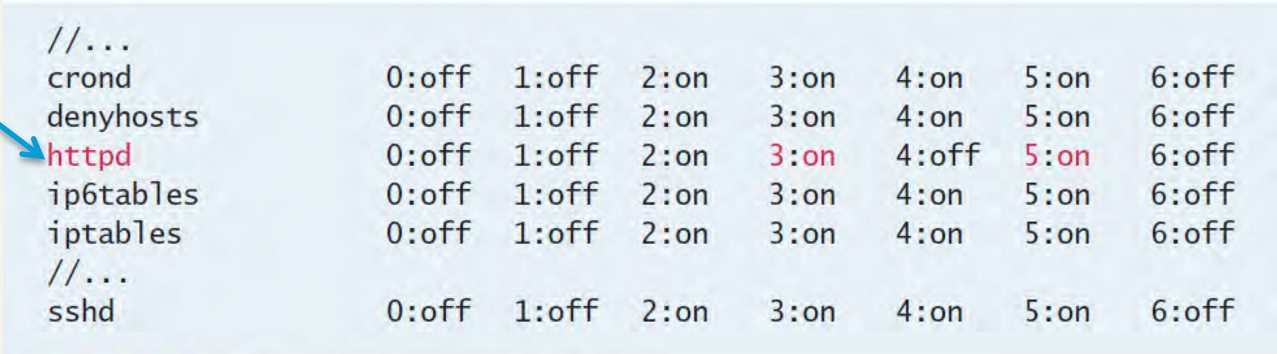
Make sure it starts on boot

You can check to see what is running on boot by typing:

chkconfig –list

The output will show the daemon name and a run level 0–6

httpd is apache



```
//...
crond              0:off  1:off  2:on   3:on   4:on   5:on   6:off
denyhosts          0:off  1:off  2:on   3:on   4:on   5:on   6:off
httpd              0:off  1:off  2:on   3:on   4:off  5:on   6:off
ip6tables          0:off  1:off  2:on   3:on   4:on   5:on   6:off
iptables           0:off  1:off  2:on   3:on   4:on   5:on   6:off
//...
sshd               0:off  1:off  2:on   3:on   4:on   5:on   6:off
```

LISTING 19.1 Output from a chkconfig listing

Run Levels

Linux Runlevels

Linux defines multiple “levels” in which the operating system can run, which correspond to different levels of service. Although the details vary between distributions they are generally considered to be:

0. Halt (shut down)

1. Single-user mode

2. Multiuser mode, no networking

3. Multiuser mode with networking

4. Unused

5. Multiuser mode with networking and GUI (Windows)

6. Reboot

Run Levels

Linux Runlevels

In practice, we normally consider only two run levels,

- run level 3 (headless production machine)
- run level 5 (development machine with GUI)

Since many services are needed on all levels, you can easily turn on the Apache daemon for levels 2, 3, 4, and 5 at boot by typing the command:

```
chkconfig httpd on
```

Similarly, to turn off an **FTP** service one can type the command:

```
chkconfig ftpd off
```

Applying configuration changes

Restarting Apache

Every time you make a change to a configuration file, you must **restart** the daemon in order for the changes to take effect.

```
/etc/init.d/httpd restart
```

However, if there’s an error in your configuration file, the server will stop, and then not restart!

Always check your configuration before restarting using:

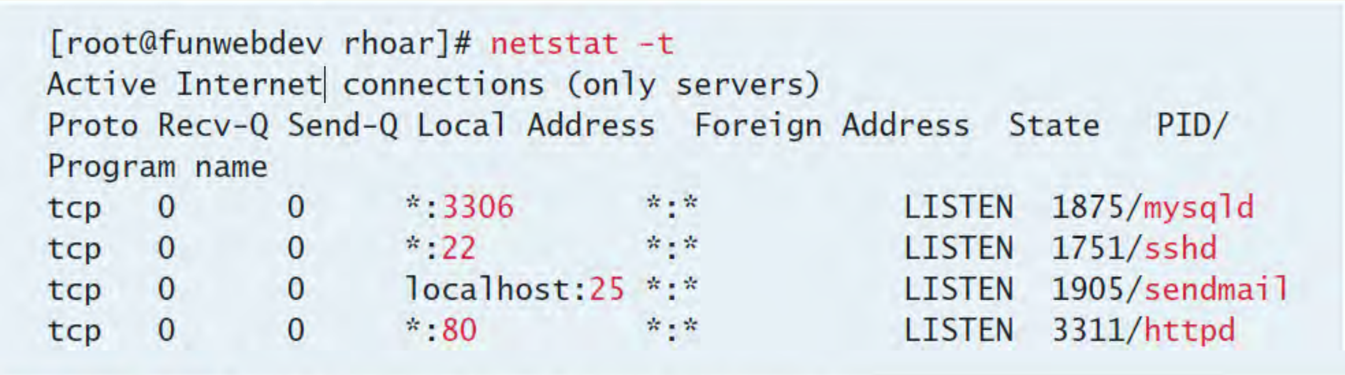
```
/etc/init.d/httpd configtest
```

This command will literally output *Syntax OK*

Connection Management

And observation

The **netstat -t** command shows which daemons are running and listening to network ports



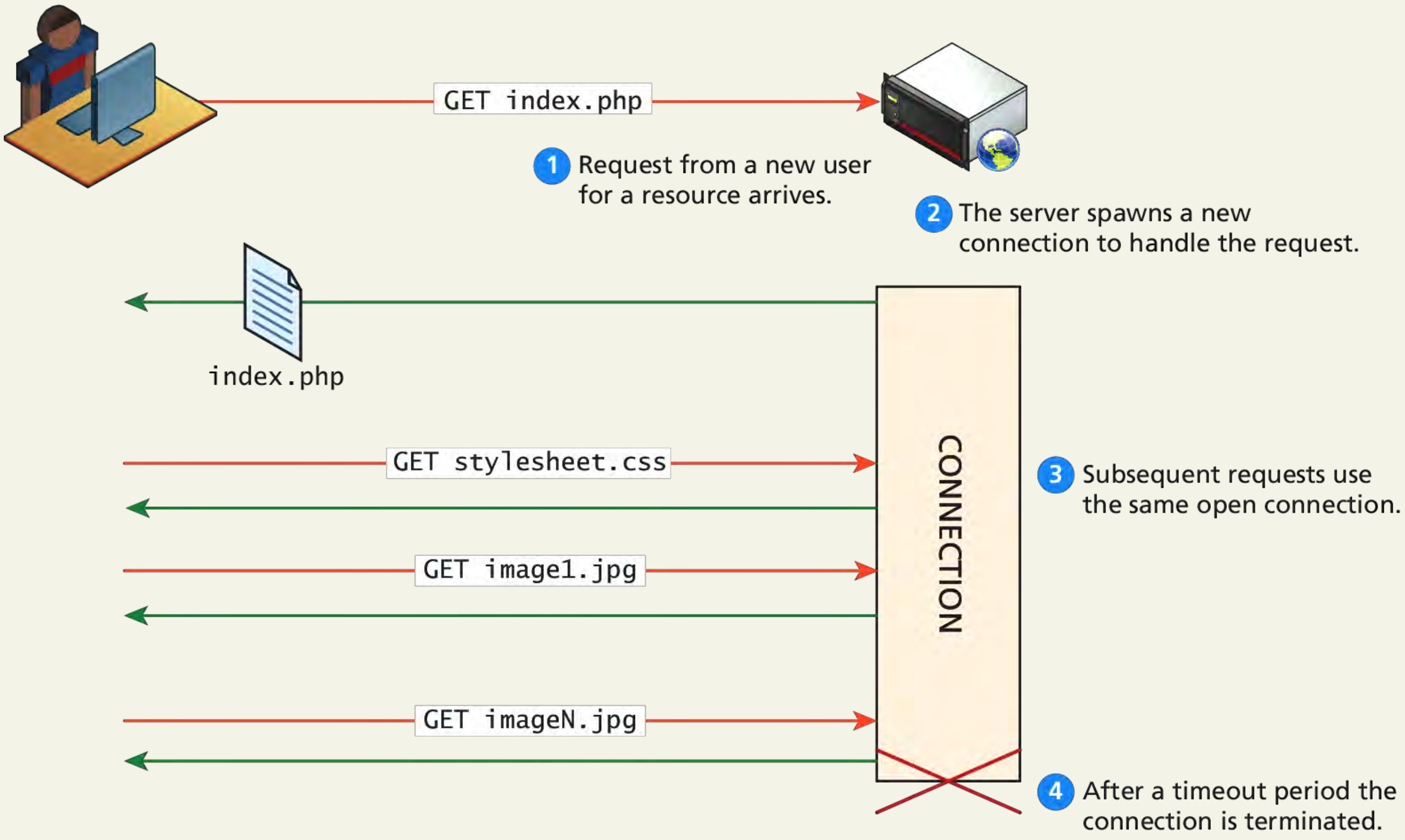
```
[root@funwebdev rhoar]# netstat -t
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address   Foreign Address   State   PID/
Program name
tcp    0      0      *:3306         *:*               LISTEN  1875/mysqld
tcp    0      0      *:22           *:*               LISTEN  1751/sshd
tcp    0      0      localhost:25   *:*               LISTEN  1905/sendmail
tcp    0      0      *:80           *:*               LISTEN  3311/httpd
```

LISTING 19.2 Sample output from a netstat command

In addition to being aware of which services are listening in general, you can manage numerous configuration options related to the number and type of connections for Apache.

Connection Management

Fine tuning your server


GET index.php
1 Request from a new user for a resource arrives.
2 The server spawns a new connection to handle the request.
index.php
GET stylesheet.css
CONNECTION
3 Subsequent requests use the same open connection.
GET image1.jpg
GET imageN.jpg
4 After a timeout period the connection is terminated.

Connection Management

Fine tuning your server

These options permit a detailed tuning of your server for various loads using **configuration directives** stored in the Apache configuration files.

- **Timeout** defines how long, in seconds, the server waits for receipts from the client (remember, delivery is guaranteed).
- **KeepAlive** is a Boolean value that tells Apache whether or not to allow more than one request per connection.
- **MaxKeepAliveRequests** sets how many requests to allow per persistent connection.
- **KeepAliveTimeout** tells the server how long to keep a connection alive between requests.

Connection Management

Fine tuning your server

It’s a balancing act with no single solution.

- Open connections take resources that could go toward serving new requests
- Allowing multiple requests from the same client to be served by the same connection saves resources by not having to spawn a new connection for each request

Additional directives like **StartServers**, **MaxClients**, **MaxRequestsPerChild**, and **ThreadsPerChild** provide additional control over the number of threads, processes, and connections per thread.

Ports

Listen

In Apache terminology, the server is said to *listen* for requests on specific *ports*.

Recall that the various TCP/IP protocols are assigned port numbers. For instance,

- the FTP protocol is assigned port 21, while
- the HTTP protocol is assigned port 80

In Apache, the Listen directive tells the server which IP/Port combinations to listen on.

```
Listen 80
```

If you want to have websites on different ports, you can use multiple Listen directives.

Data Compression

Saving bandwidth

The HTTP headers allow client and server to know whether compression can be used.

Deciding whether to compress data may at first glance seem like an easy decision but some files like .jpg files are already compressed, and re-compressing them will use up CPU time needlessly.

The Apache directive below adds compression (when agreed to with the client) to items of type text/html

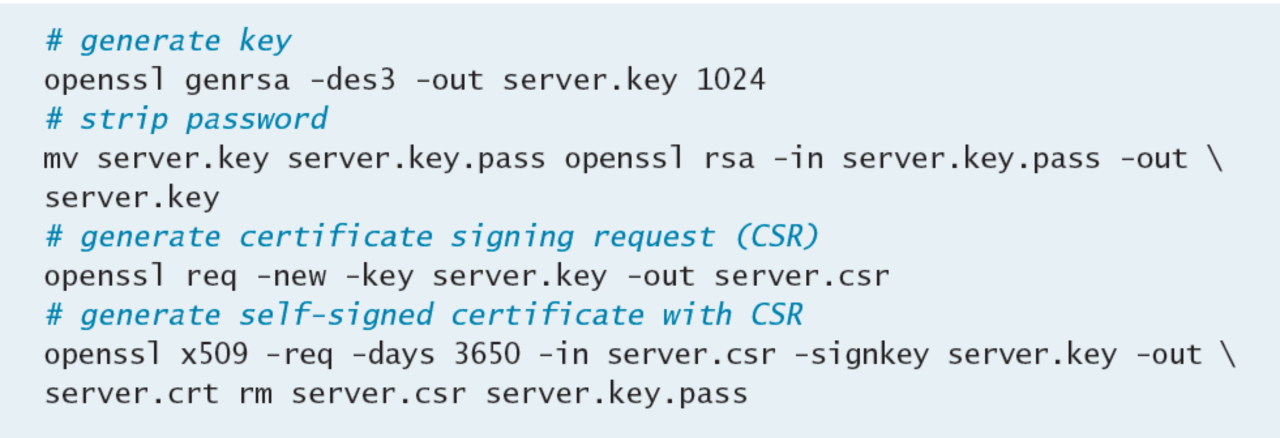
```
AddOutputFilterByType DEFLATE text/html
```

Encryption and SSL

Remember the cryptography from Chapter 16?

All encrypted traffic requires the use of an X.509 public key certificate, which contains cryptographic keys as well as information about the site (identity).

creating your own certificates is very straightforward, as illustrated by the shell script below



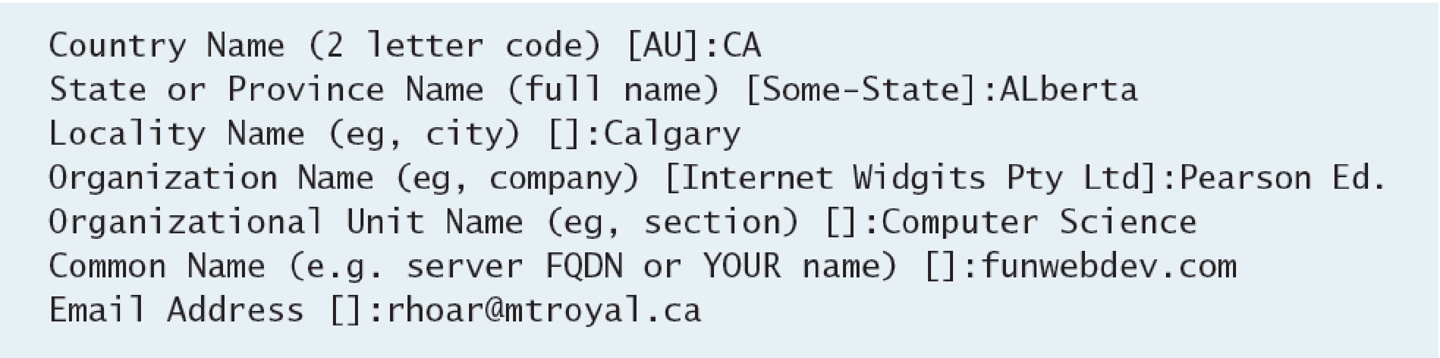
```
# generate key
openssl genrsa -des3 -out server.key 1024
# strip password
mv server.key server.key.pass openssl rsa -in server.key.pass -out \
server.key
# generate certificate signing request (CSR)
openssl req -new -key server.key -out server.csr
# generate self-signed certificate with CSR
openssl x509 -req -days 3650 -in server.csr -signkey server.key -out \
server.crt rm server.csr server.key.pass
```

LISTING 19.3 Script to generate a self-signed certificate

Encryption and SSL

Certificate Signing

Self-signed certificates work; it’s just that the user will have to approve an exception to the strict rules configured by most browsers.



```
Country Name (2 letter code) [AU]:CA
State or Province Name (full name) [Some-State]:ALberta
Locality Name (eg, city) []:Calgary
Organization Name (eg, company) [Internet Widgits Pty Ltd]:Pearson Ed.
Organizational Unit Name (eg, section) []:Computer Science
Common Name (e.g. server FQDN or YOUR name) []:funwebdev.com
Email Address []:rhoar@mtroyal.ca
```

LISTING 19.4 Questions and answers to generate the certificate-signing request

Encryption and SSL

Apache configuration details

Signed certificates generally require uploading the certificate signing request generated in Listing 19.3 to get a **server.crt** file returned by email.

However you sign, you will have two files that are used by Apache

SSLCertificateFile /path/to/this/**server.crt**

SSLCertificateKeyFile /path/to/this/**server.key**

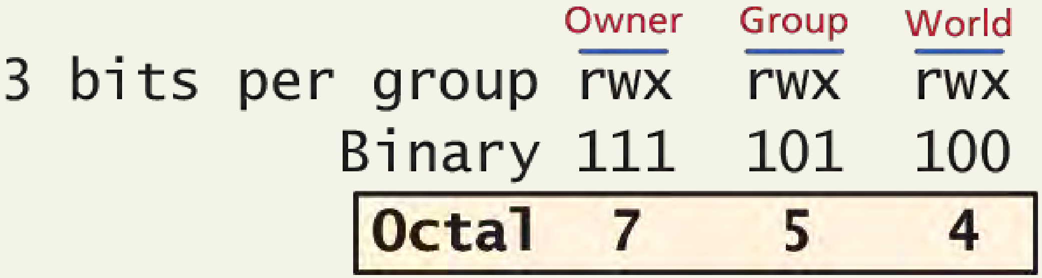
Remember, you must also ***Listen*** on port **443** in order to get Apache to work correctly using secure connections.

File Ownership and Permissions

A review for many

Apache runs as its own user (sometimes called Apache, WWW, or HTTP depending on configuration). In to serve files, Apache needs permission to access them.

Typically, newly created PHP files are granted 644 octal permissions so that the owner can read and write, while the group and world can read. This means that no matter what username Apache is running under, it can read the file.



	Owner	Group	World
3 bits per group	rwx	rwx	rwx
Binary	111	101	100
Octal	**7**	**5**	**4**

File Ownership and Permissions

Security risk

A security risk can arise on a shared server if you set a file to world writable.

This means users on the system who can get access to that file can write their own content to it, circumventing any authentication you have in place.

Many shared hosts have been “hacked” by a user simply overwriting the **index.php** file with a file of their choosing.

This is why you should never set permissions to **777**, especially on a simple shared host.

Section 4 of 5

APACHE REQUEST AND RESPONSE MANAGEMENT

Managing Multiple Domains

On One Webserver

A web server can easily be made to serve multiple sites from the same machine.

Having multiple sites running on a single server can be a great advantage to companies or individuals hosting multiple small websites.

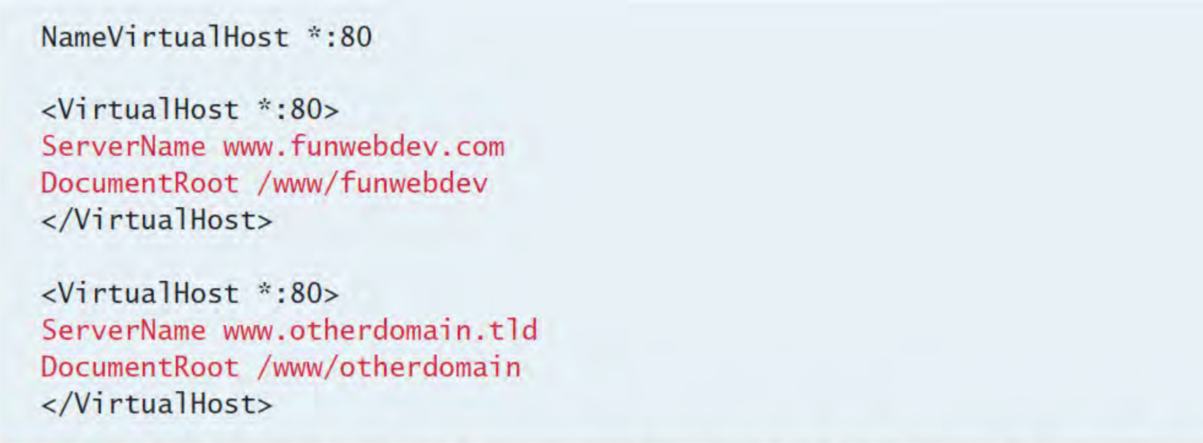
A **VirtualHost** is an Apache configuration directive that associates a particular combination of server name and port to a folder on the server.

Managing Multiple Domains

VirtualHost Directive

Each distinct **VirtualHost** must specify

- which IP and port to listen on
- what file system location to use as the root for that domain.
- NameVirtualHost allows you to use domain names instead of IP addresses. This means many domains on 1 IP address!

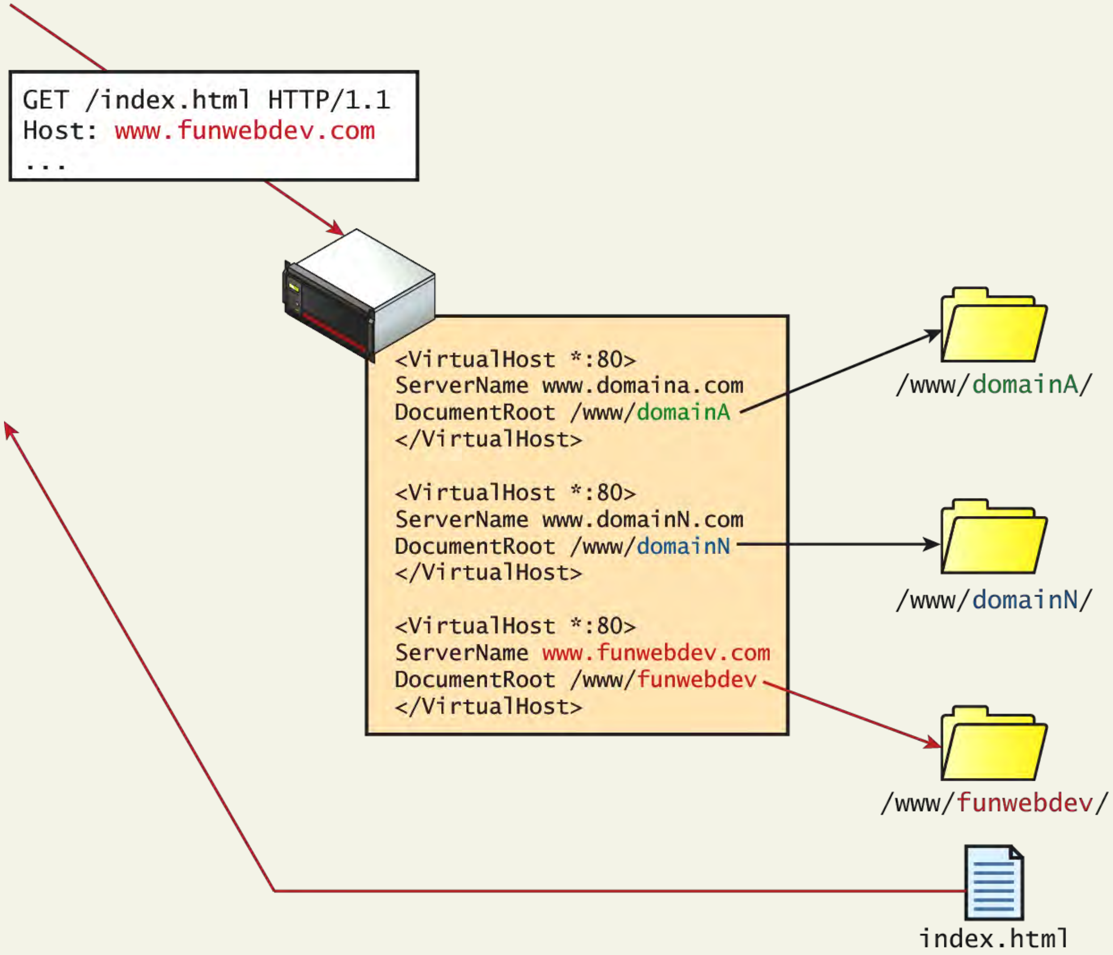


```
NameVirtualHost *:80

<VirtualHost *:80>
ServerName www.funwebdev.com
DocumentRoot /www/funwebdev
</VirtualHost>

<VirtualHost *:80>
ServerName www.otherdomain.tld
DocumentRoot /www/otherdomain
</VirtualHost>
```

LISTING 19.5 Apache VirtualHost directives in httpd.conf for two different domains on same IP address

Managing Multiple Domains

VirtualHost Visualization



```
GET /index.html HTTP/1.1
Host: www.funwebdev.com
...
```

```
<VirtualHost *:80>
ServerName www.domaina.com
DocumentRoot /www/domainA
</VirtualHost>

<VirtualHost *:80>
ServerName www.domainN.com
DocumentRoot /www/domainN
</VirtualHost>

<VirtualHost *:80>
ServerName www.funwebdev.com
DocumentRoot /www/funwebdev
</VirtualHost>
```

Handling Directory Requests

The index files

In practice, users normally request a domain’s homepage URL without specifying what file they want.

There are times when clients are requesting a folder path, rather than a file path. The domain root is a special case of the folder question, where the folder being requested is the root folder for that domain.

However a folder is requested, the server must be able to determine what to serve in response

Handling Directory Requests

What to serve?

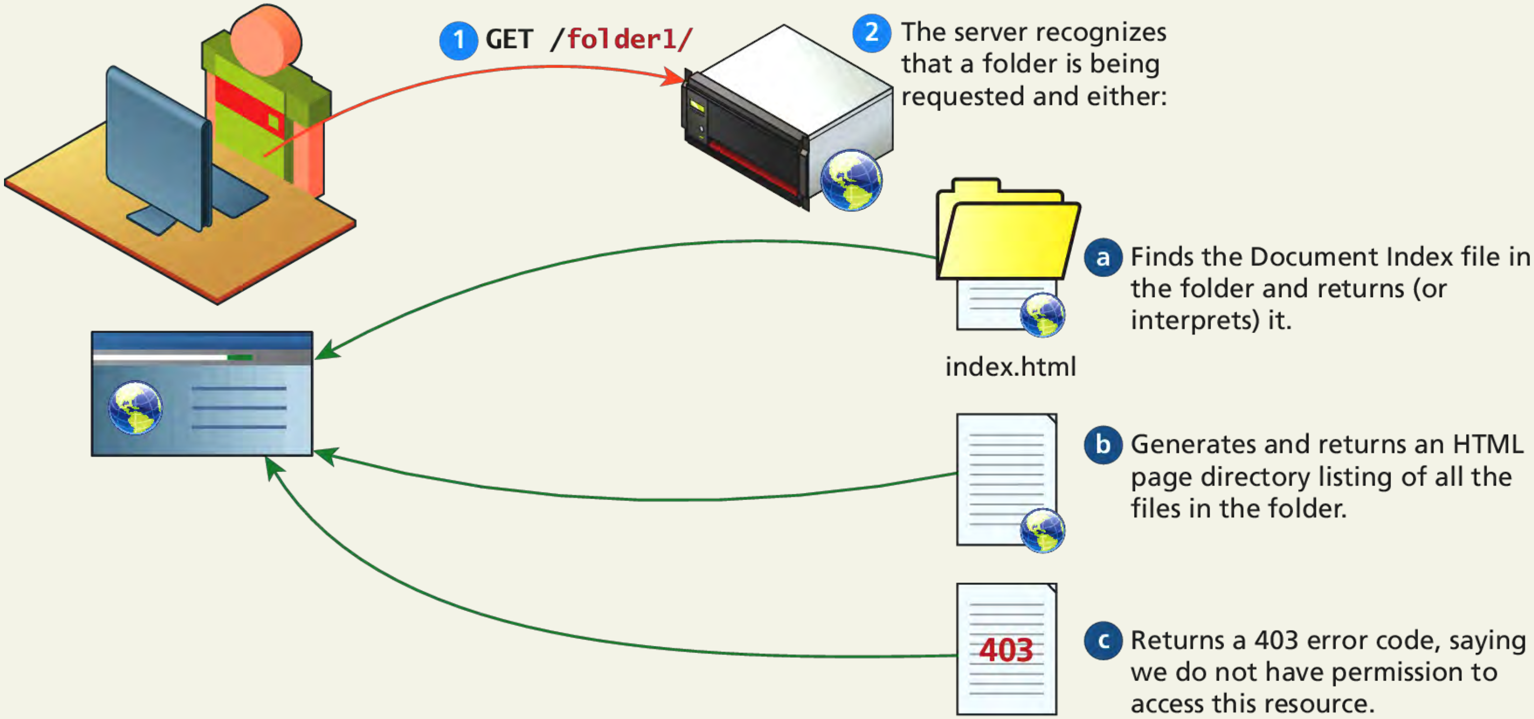
The server could choose

- a file to serve
- display the directory contents
- return an error code

You can control this by adding **DirectoryIndex** and **Options** directives to the Apache configuration file.

Handling Directory Requests

What to serve?



1 GET /folder1/

2 The server recognizes that a folder is being requested and either:

a Finds the Document Index file in the folder and returns (or interprets) it.

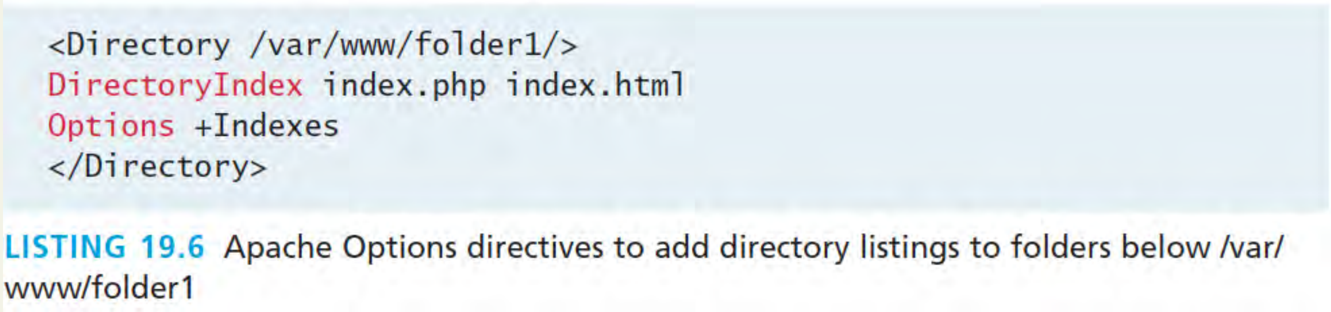
b Generates and returns an HTML page directory listing of all the files in the folder.

c Returns a 403 error code, saying we do not have permission to access this resource.

Handling Directory Requests

How did it come to pass that we use index.php

The **DirectoryIndex** directive configures the server to respond with a particular file



```
<Directory /var/www/folder1/>
DirectoryIndex index.php index.html
Options +Indexes
</Directory>
```

LISTING 19.6 Apache Options directives to add directory listings to folders below /var/www/folder1

in this case **index.php**, and if it’s not present, **index.html**

The **Options** directives can be used to tell the server to build a clickable index page from the content of the folder in response to a folder request.

Responding to File Requests

Static and Dynamic

The most basic operation a web server performs is responding to an HTTP request for a **static** file.

Having mapped the request to a particular file location using the connection management options above, the server sends the requested file, along with the relevant HTTP headers to signify that this request was successfully responded to.

dynamic file requests must be interpreted at request time rather than sent back directly as responses

Responding to File Requests

Which files get interpreted

A web server associates certain file extensions with MIME types that need to be interpreted. When you install Apache for PHP, this is done automatically, but can be overridden through directives.

If you wanted files with PHP as well as HTML extensions to be interpreted (so you could include PHP code inside them), you would add the directive below, which uses the PHP MIME types:

```
AddHandler application/x-httpd-php .php

AddHandler application/x-httpd-php .html
```

URL Redirection

We've come across this before…

In Apache, there are two major classes of redirection,

- **public redirection** and
- **internal redirection** (also called **URL rewriting**).

Public Redirection

In public redirection, you may have a URL that no longer exists or has been moved.

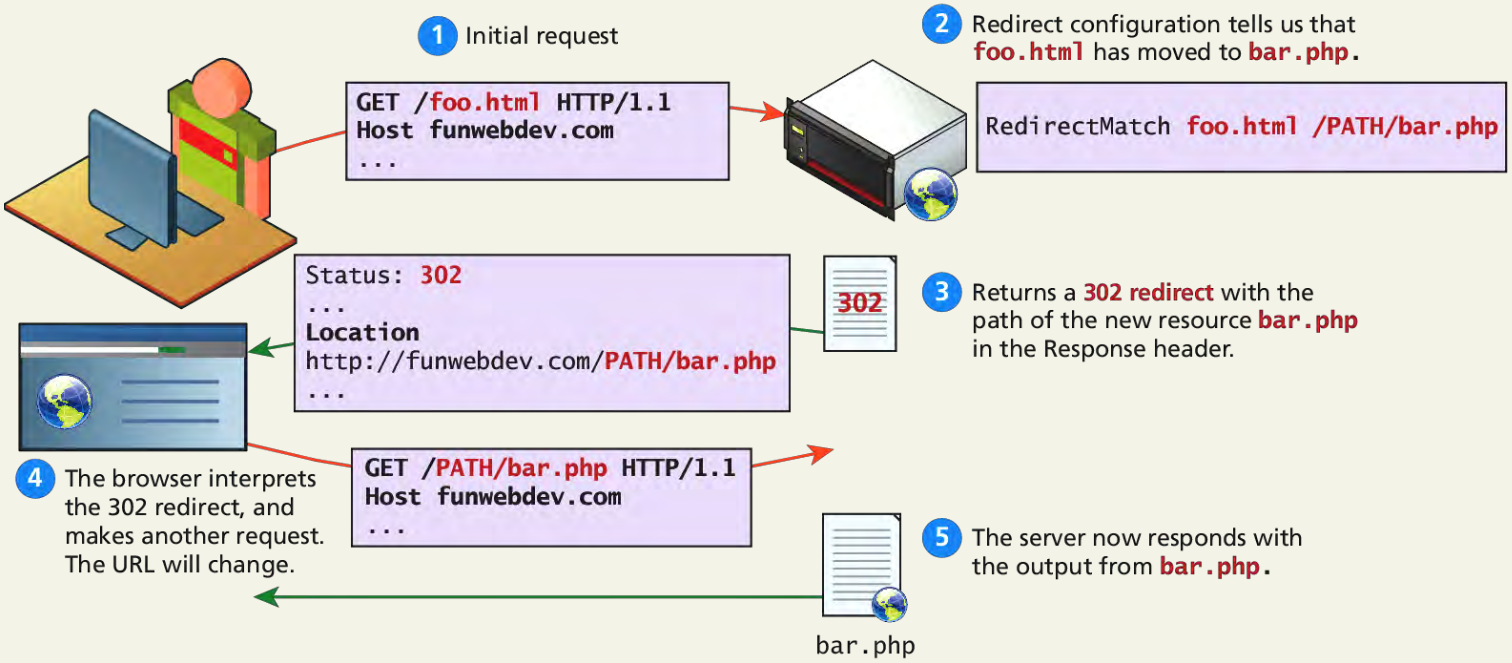
If users have bookmarks to old URLs, they will get **404** error codes when requesting them

It is a better practice to inform users that their old pages have moved, using a HTTP **302** header

In Apache such URL redirection is easily achieved, using Apache directives

Public Redirection

Two requests required, and everybody knows



1 Initial request

```
GET /foo.html HTTP/1.1
Host funwebdev.com
...
```

2 Redirect configuration tells us that **foo.html** has moved to **bar.php**.

```
RedirectMatch foo.html /PATH/bar.php
```

```
Status: 302
...
Location
http://funwebdev.com/PATH/bar.php
...
```

3 Returns a **302 redirect** with the path of the new resource **bar.php** in the Response header.

4 The browser interprets the 302 redirect, and makes another request. The URL will change.

```
GET /PATH/bar.php HTTP/1.1
Host funwebdev.com
...
```

5 The server now responds with the output from **bar.php**.

Public Redirection

There are Apache Directives

Using **RedirectMatch** foo.html is publically redirected to bar.php

```
RedirectMatch /foo.html /FULLPATH/bar.php
```

Alternatively the **RewriteEngine** module can be invoked to create an equivalent rule:

```
RewriteEngine on
RewriteRule ^/foo\.html$ /FULLPATH/bar.php [R]
```

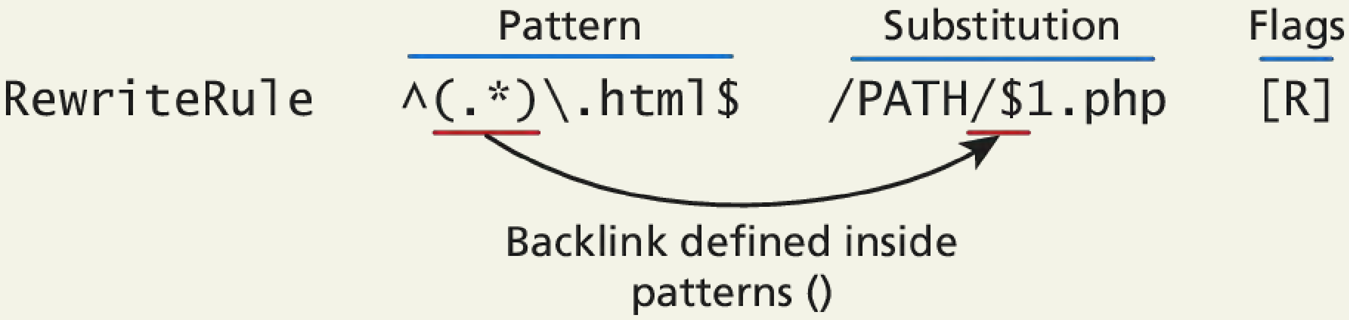
Public Redirection

The RedirectRule Directive

RewriteRule directive consists of three parts:

- the pattern to match,
- the substitution, and
- Flags

Use can use **regular expression syntax** to capture back-references for use in the substitution.



Pattern

Substitution

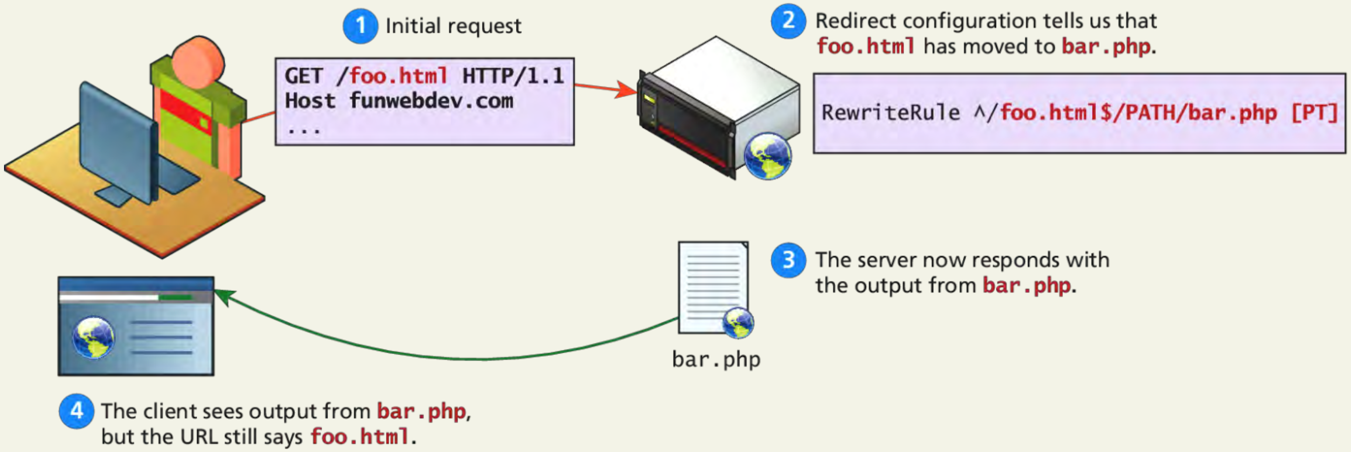
Flags

```
RewriteRule   ^(.*)\.html$   /PATH/$1.php   [R]
```

Backlink defined inside
patterns ()

Internal Redirection

One fewer requests



1 Initial request

```
GET /foo.html HTTP/1.1
Host funwebdev.com
...
```

2 Redirect configuration tells us that **foo.html** has moved to **bar.php**.

```
RewriteRule ^/foo.html$/PATH/bar.php [PT]
```

3 The server now responds with the output from **bar.php**.

4 The client sees output from **bar.php**, but the URL still says **foo.html**.

Internal Redirection

One fewer requests

To enable such a case, simply modify the rewrite rule’s flag from redirect (R) to pass-through (PT), which indicates to pass-through internally and not redirect.

```
RewriteEngine on

RewriteRule ^/foo\.html$ /FULLPATH/bar.php [PT]
```

Conditional ReWriting

Internal or Public

RewriteCondition combined with the **RewriteRule** can be thought of as a conditional statement.

If more than one rewrite condition is specified, they must all match for the rewrite to execute.

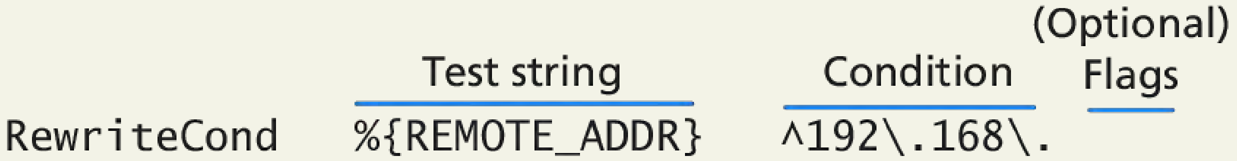
The RewriteCond consists of three parts,

- a test string
- and a conditional pattern.
- Sometimes flags, is also used.

Conditional ReWriting

Internal or Public

The example below allows us to redirect if the request is coming from an IP that begins with 192.168.



```
RewriteCond    %{REMOTE_ADDR}    ^192\.168\.
```

Conditional ReWriting

An advanced example

To prevent **hot-linking** of your image files consider a conditional redirect that only allows images to be returned if the HTTP_REFERER header is from our domain:

NC – Case insensitive

```
RewriteEngine On
RewriteCond %{HTTP_REFERER} !^http://(www\.)? funwebdev\.com/.*$ [NC]
RewriteRule \.(jpg|gif|bmp|png)$ - [F]
```

F - Forbidden

To return a small static image for all invalid requests use the following directives:

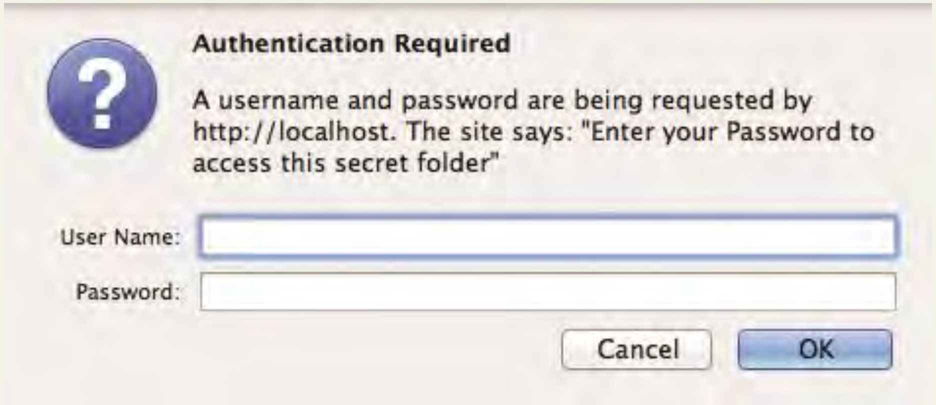
```
RewriteEngine On
RewriteCond %{HTTP_REFERER} !^http://(www\.)?funwebdev\.com/.*$ [NC]
RewriteRule \.(jpg|gif|bmp|png)$ http://funwebdev.com/stopIt.png
```

Managing Access with .htaccess

Should have done his a long time ago (maybe you did)

.htaccess files are the directory-level configuration files used by Apache to store directives to apply to this particular folder.

While most websites will track and manage users using a database with PHP authentication scripts, a simpler mechanism exists when you need to quickly password protect a file or folder.



Authentication Required

A username and password are being requested by http://localhost. The site says: "Enter your Password to access this secret folder"

User Name:

Password:

Managing Access with .htaccess

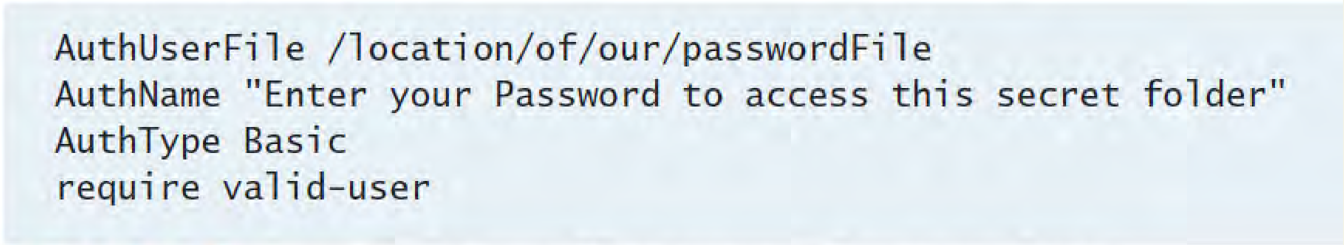
Add a file to the folder and point to a password file

To create a new password file, you would type the following command:

htpasswd –c passwordFile ricardo

This will create a file named *passwordFile* and prompt you for a password for the user *ricardo* (I chose *password*).

.htaccess, can now point to that password file



```
AuthUserFile /location/of/our/passwordFile
AuthName "Enter your Password to access this secret folder"
AuthType Basic
require valid-user
```

LISTING 19.8 A sample .htaccess file to password protect a folder

Server Caching

Another Cache

Server caching is distinct from:

- **HTTP caching** built into the HTTP protocol
- the caching technique using PHP described in Chapter 13

Apache caching supplements provides another caching mechanism (in the form of a module, mod_cache) that allows you to save copies of HTTP responses on the server so that the PHP script that created them won't have to run again.

Server Caching

Another Cache

There are two types of server cache,

- a **memory cache**
- a **disk cache**.

The memory cache is faster, but of course the server RAM is limited. The disk cache is slower, but can support more data.

Caching is based on URLs so that every cached page is associated with a particular URL.

Server Caching

Directives – in brief

Some important directives related to the mod_cache module are:

- **CacheEnable** turns caching on. You include whether to use *disk* or *memory* caching and the location. To cache all requests for a subdomain **archive.funwebdev.com**, you would type the directive.

  CacheEnable disk archive.funwebdev.com

- **CacheRoot** defines the folder on your server to store all the cached resources. You might save cached files in a high-speed, solid-state mounted disk, for instance, as follows:

  CacheRoot /fastdisk/cache/

- **CacheDefaultExpire** determines how long in seconds something in cache is stored before the cached copy expires.

Section 5 of 5

WEB MONITORING AND ANALYTICS

Monitoring

Internal and External

Internal monitoring reads the outputted logs of all the daemons to look for potential issues.

External monitoring is installed off of the server and checks to see that connections to required services are open.

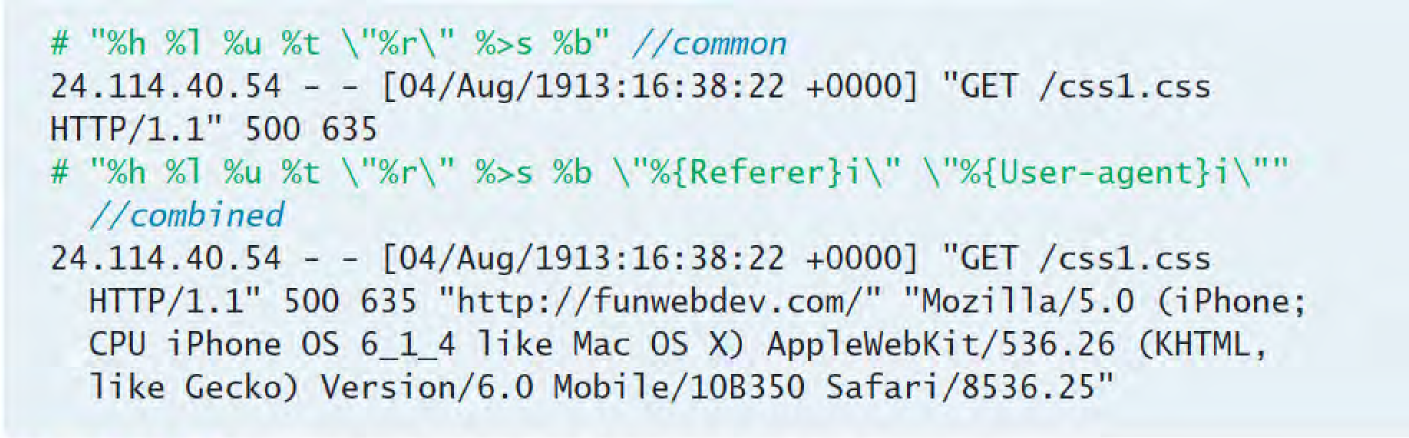
Internal Monitoring

Apache Logging

Logging relates closely to Apache, since Apache directives determine what information goes into the WWW logs.

You can define a log file using the directive CustomLog:

CustomLog /var/log/funwebdev/access_log *nickname*



```
# "%h %l %u %t \"%r\" %>s %b" //common
24.114.40.54 - - [04/Aug/1913:16:38:22 +0000] "GET /css1.css
HTTP/1.1" 500 635
# "%h %l %u %t \"%r\" %>s %b \"%{Referer}i\" \"%{User-agent}i\""
  //combined
24.114.40.54 - - [04/Aug/1913:16:38:22 +0000] "GET /css1.css
  HTTP/1.1" 500 635 "http://funwebdev.com/" "Mozilla/5.0 (iPhone;
  CPU iPhone OS 6_1_4 like Mac OS X) AppleWebKit/536.26 (KHTML,
  like Gecko) Version/6.0 Mobile/10B350 Safari/8536.25"
```

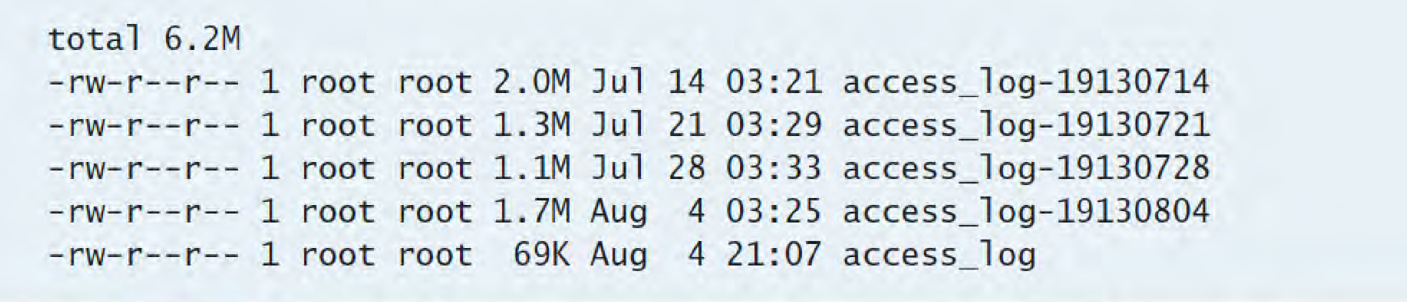
LISTING 19.9 Sample log formats and example outputs

Internal Monitoring

Log rotation

If no maintenance of your log files is ever done, then the logs would keep accumulating and the file would grow in size until eventually it would start to impact performance or even use up all the space on the system.

logrotate is the daemon running on most systems by default to handle this task.



```
total 6.2M
-rw-r--r-- 1 root root 2.0M Jul 14 03:21 access_log-19130714
-rw-r--r-- 1 root root 1.3M Jul 21 03:29 access_log-19130721
-rw-r--r-- 1 root root 1.1M Jul 28 03:33 access_log-19130728
-rw-r--r-- 1 root root 1.7M Aug  4 03:25 access_log-19130804
-rw-r--r-- 1 root root  69K Aug  4 21:07 access_log
```

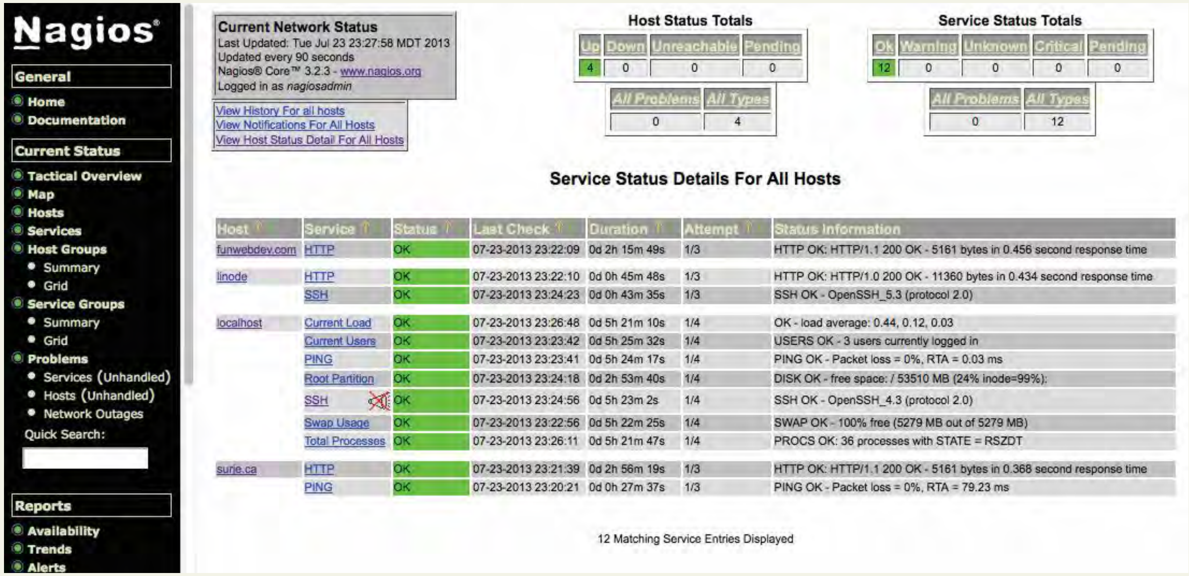
LISTING 19.10 Output of the ls -lrt command in a log folder showing log rotation

External Monitoring

Test the network

Monitoring software like **Nagios** can check for uptime and immediately notify the administrator if a service goes down.

Much like internal logs, external monitoring logs can be used to generate uptime reports and other visual summaries of your server.



Current Network Status
Last Updated: Tue Jul 23 23:27:58 MDT 2013
Updated every 90 seconds
Nagios® Core™ 3.2.3 - www.nagios.org
Logged in as *nagiosadmin*

View History For all hosts
View Notifications For All Hosts
View Host Status Detail For All Hosts

Host Status Totals

Up	Down	Unreachable	Pending
4	0	0	0

All Problems	*All Types*
0	4

Service Status Totals

Ok	Warning	Unknown	Critical	Pending
12	0	0	0	0

All Problems	*All Types*
0	12

Service Status Details For All Hosts

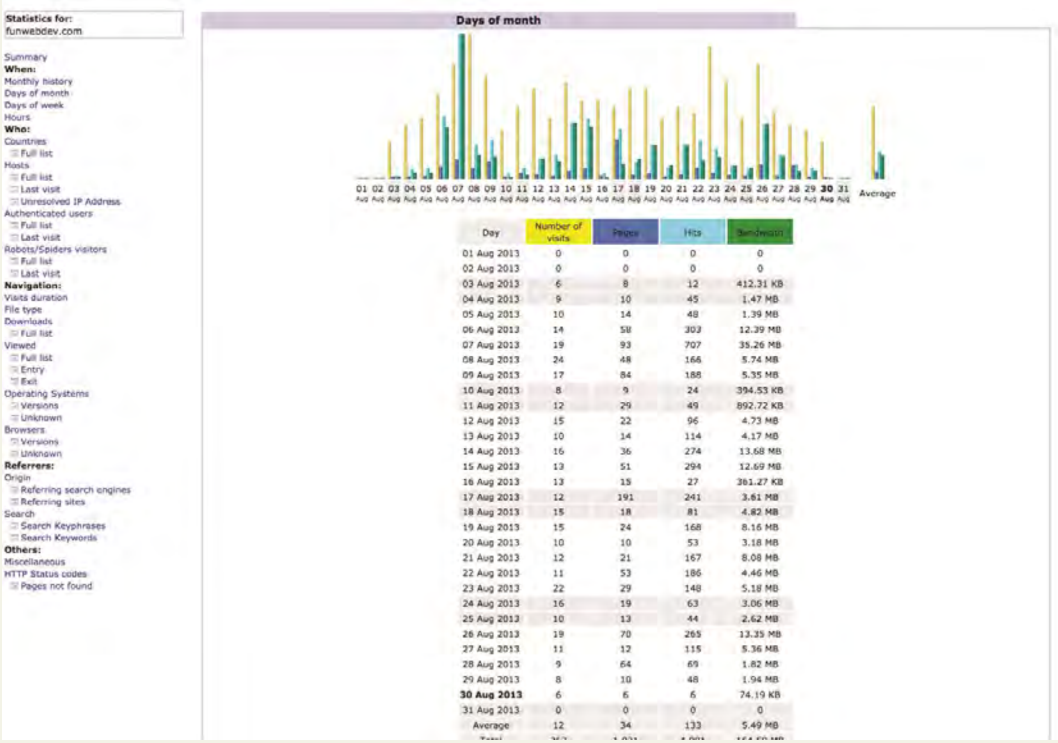
Host	Service	Status	Last Check	Duration	Attempt	Status Information
funwebdev.com	HTTP	OK	07-23-2013 23:22:09	0d 2h 15m 49s	1/3	HTTP OK: HTTP/1.1 200 OK - 5161 bytes in 0.456 second response time
linode	HTTP	OK	07-23-2013 23:22:10	0d 0h 45m 48s	1/3	HTTP OK: HTTP/1.0 200 OK - 11360 bytes in 0.434 second response time
	SSH	OK	07-23-2013 23:24:23	0d 0h 43m 35s	1/3	SSH OK - OpenSSH_5.3 (protocol 2.0)
localhost	Current Load	OK	07-23-2013 23:26:48	0d 5h 21m 10s	1/4	OK - load average: 0.44, 0.12, 0.03
	Current Users	OK	07-23-2013 23:23:42	0d 5h 25m 32s	1/4	USERS OK - 3 users currently logged in
	PING	OK	07-23-2013 23:23:41	0d 5h 24m 17s	1/4	PING OK - Packet loss = 0%, RTA = 0.03 ms
	Root Partition	OK	07-23-2013 23:24:18	0d 2h 53m 40s	1/4	DISK OK - free space: / 53510 MB (24% inode=99%):
	SSH	OK	07-23-2013 23:24:56	0d 5h 23m 2s	1/4	SSH OK - OpenSSH_4.3 (protocol 2.0)
	Swap Usage	OK	07-23-2013 23:22:56	0d 5h 22m 25s	1/4	SWAP OK - 100% free (5279 MB out of 5279 MB)
	Total Processes	OK	07-23-2013 23:26:11	0d 5h 21m 47s	1/4	PROCS OK: 36 processes with STATE = RSZDT
surje.ca	HTTP	OK	07-23-2013 23:21:39	0d 2h 56m 19s	1/3	HTTP OK: HTTP/1.1 200 OK - 5161 bytes in 0.368 second response time
	PING	OK	07-23-2013 23:20:21	0d 0h 27m 37s	1/3	PING OK - Packet loss = 0%, RTA = 79.23 ms

12 Matching Service Entries Displayed

Internal Analytics

Build on your logs

Analysis packages such as **AWStats** and **Webalizer** allow you to easily set up periodic analysis of the log files to create bar graphs; pie charts; and lists of top users, browsers, countries, and more



Day	Number of visits	Pages	Hits	Bandwidth
01 Aug 2013	0	0	0	0
02 Aug 2013	0	0	0	0
03 Aug 2013	6	8	12	412.31 KB
04 Aug 2013	9	10	45	1.47 MB
05 Aug 2013	10	14	48	1.39 MB
06 Aug 2013	14	58	303	12.39 MB
07 Aug 2013	19	93	707	35.26 MB
08 Aug 2013	24	48	166	5.74 MB
09 Aug 2013	17	84	188	5.35 MB
10 Aug 2013	8	9	24	394.53 KB
11 Aug 2013	12	29	49	892.72 KB
12 Aug 2013	15	22	96	4.73 MB
13 Aug 2013	10	14	114	4.17 MB
14 Aug 2013	16	36	274	13.68 MB
15 Aug 2013	13	51	294	12.69 MB
16 Aug 2013	13	15	27	361.27 KB
17 Aug 2013	12	191	241	3.61 MB
18 Aug 2013	15	18	81	4.82 MB
19 Aug 2013	15	24	168	8.16 MB
20 Aug 2013	10	10	53	3.18 MB
21 Aug 2013	12	21	167	8.08 MB
22 Aug 2013	11	53	186	4.46 MB
23 Aug 2013	22	29	148	5.18 MB
24 Aug 2013	16	19	63	3.06 MB
25 Aug 2013	10	13	44	2.62 MB
26 Aug 2013	19	70	265	13.35 MB
27 Aug 2013	11	12	115	5.36 MB
28 Aug 2013	9	64	69	1.82 MB
29 Aug 2013	8	10	48	1.94 MB
30 Aug 2013	6	6	6	74.19 KB
31 Aug 2013	0	0	0	0
Average	12	34	133	5.49 MB
Total	[illegible]	[illegible]	[illegible]	[illegible]

Third-Party Analytics

Put in a little piece of JavaScript

Third-party systems like Google Analytics provide much of the same data, but rather than collect it from your logs, they embed a small piece of JavaScript into each page of your site.

These statistics can be more robust than the free tools, but require every visit to the site to execute another script, slowing performance.

Third-Party Support Tools

Let us help

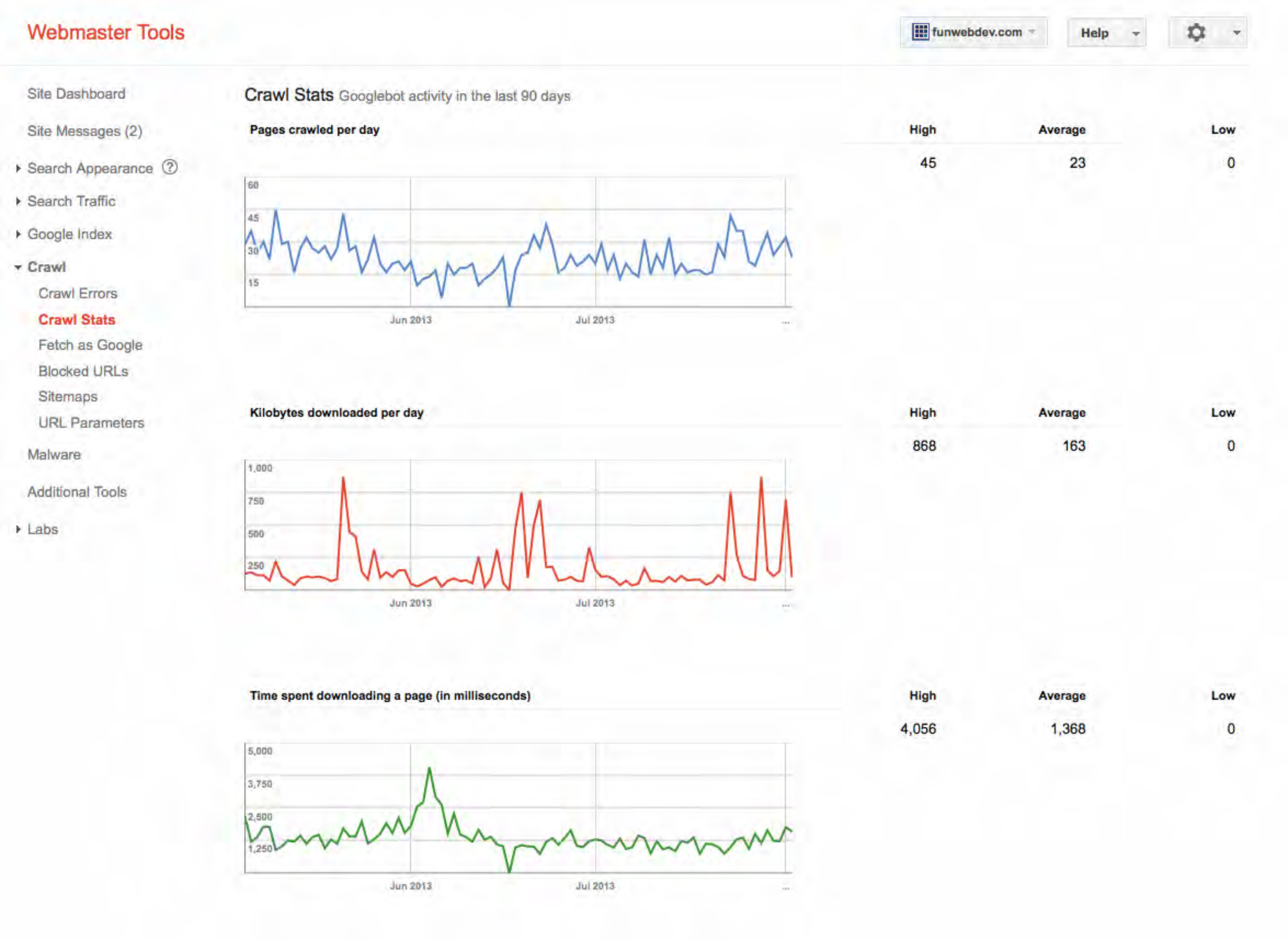
These tools provide information about

- Indexed terms and weights
- Indexing errors that were encountered
- Search ranking and traffic
- Frequency of being crawled
- Response time during the crawls

To sign up for these tools, go to **www.google.com/webmasters/tools/** and **http://www.bing.com/toolbox/webmaster**.

Third-Party Support Tools

Screenshot of Google’s Webmaster Tools



Crawl Stats Googlebot activity in the last 90 days

Pages crawled per day	High	Average	Low
	45	23	0

Kilobytes downloaded per day	High	Average	Low
	868	163	0

Time spent downloading a page (in milliseconds)	High	Average	Low
	4,056	1,368	0

What You've Learned

1 Web Server Hosting Options

2 Domain and Name Server Administration

3 Linux and Apache Configuration

4 Apache Request/Response

5 Web Monitoring and Analytics