# Web Server Administration

Chapter 19

# 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:

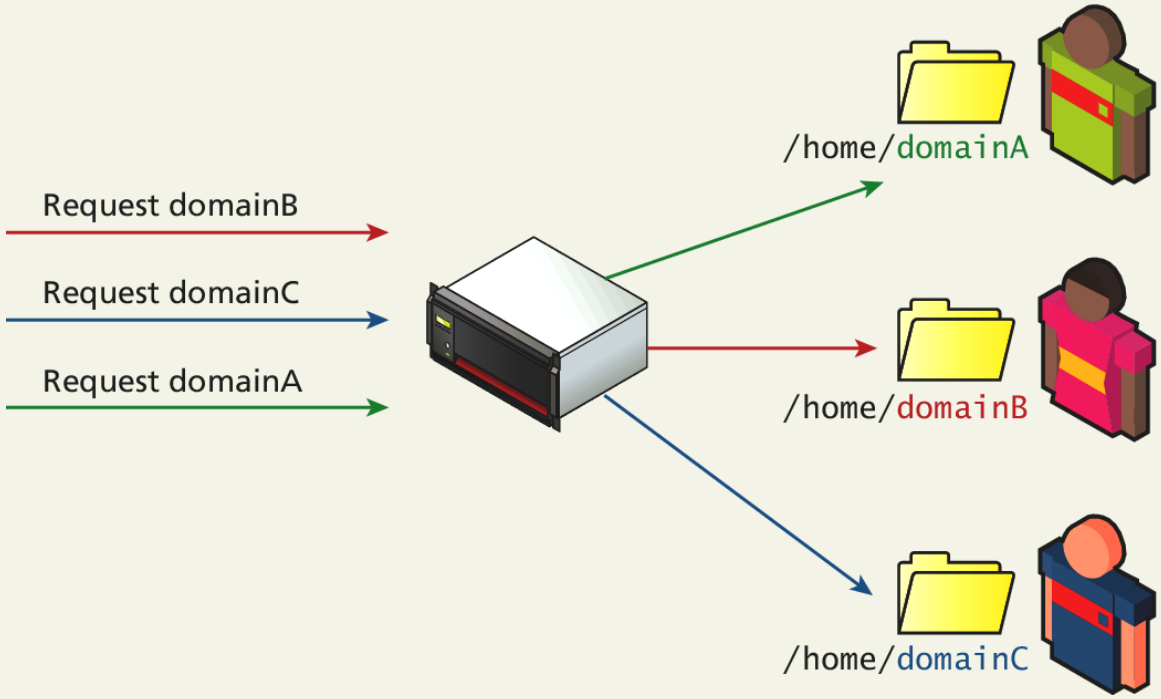
- Shared Hosting
- Collocated Hosting
- Dedicated Hosting

Not sure about that spelling

# 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…. But cheap

**The disadvantages:**

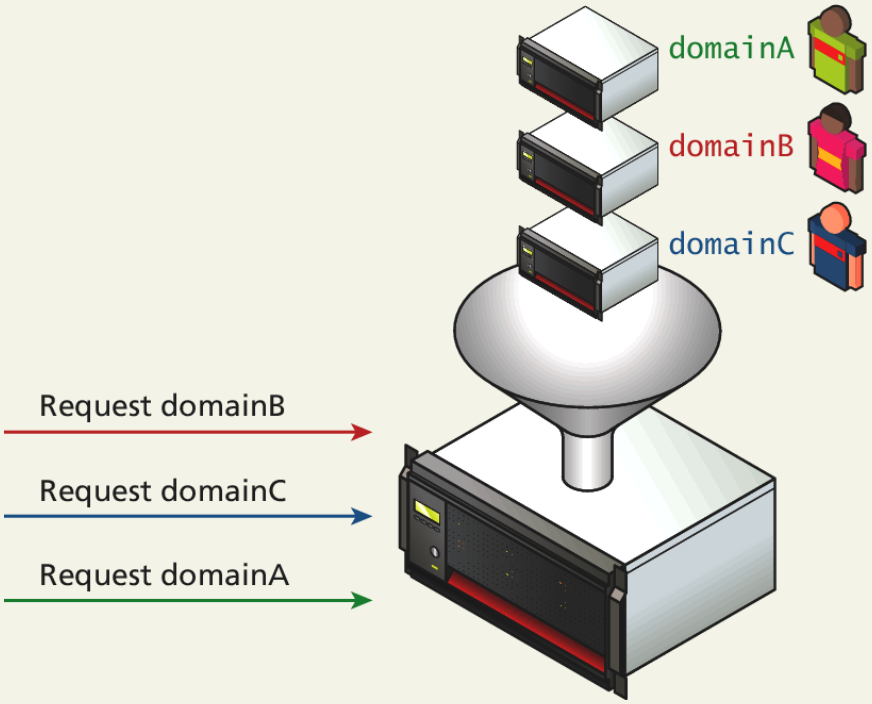
- Lack of control
- poor performance
- security threats

Looks like what we have

# 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

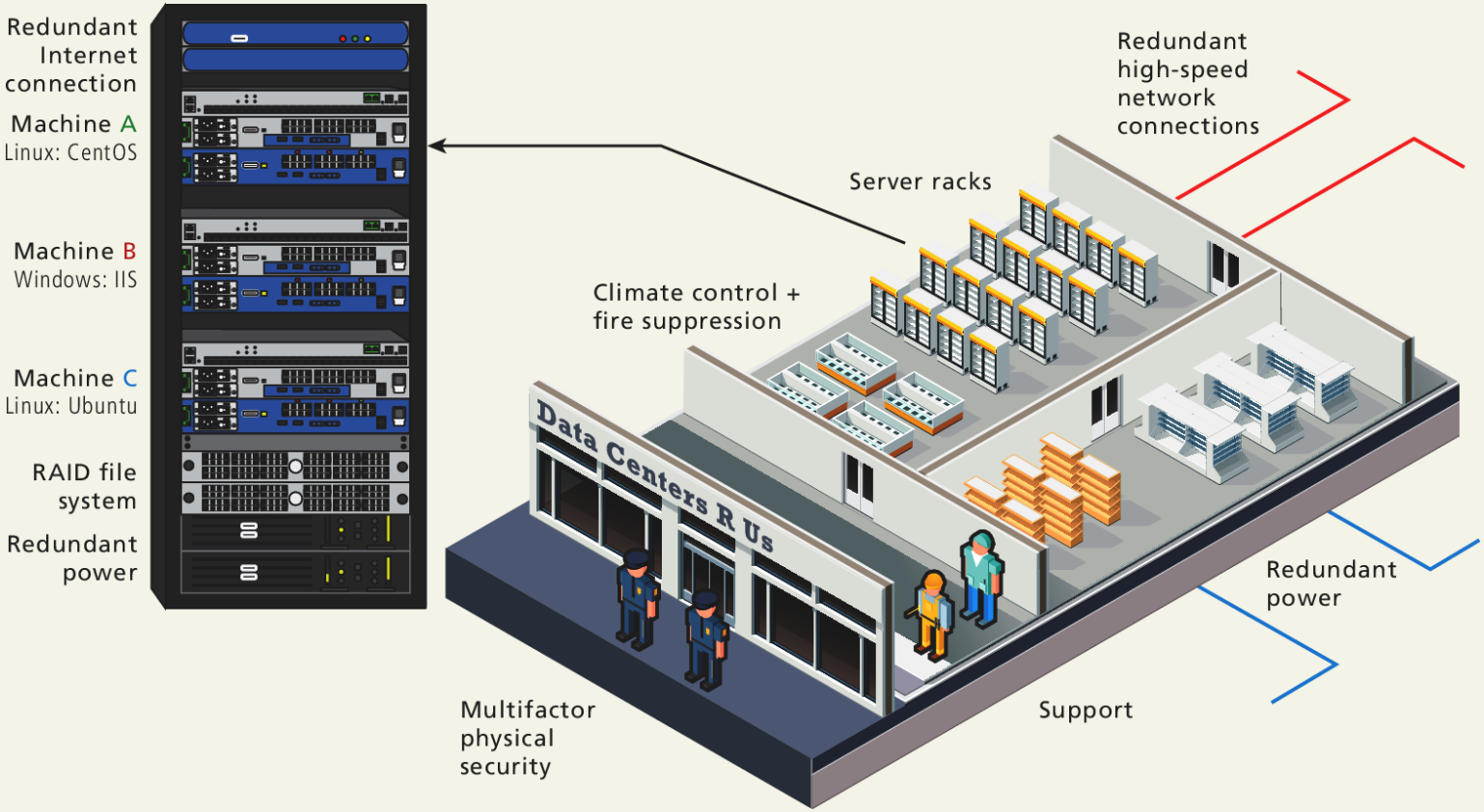



Word Press & Windows servers here probably have this

# 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. (if you want that!)

# Collocated Hosting

Touch the machine

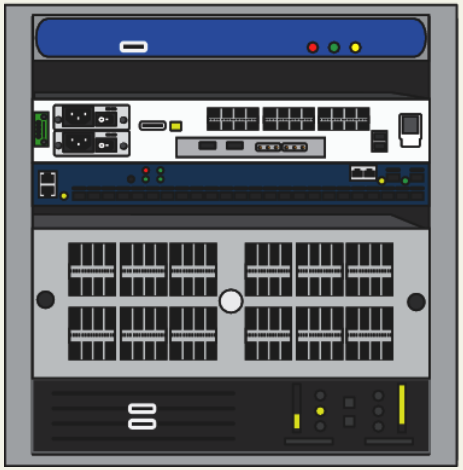
- **Collocated hosting** is almost like dedicated hosting, except rather than rent a machine, you own and manage the machine yourself.
- The advantage of collocated hosting goes beyond a dedicated server with not only full control over the OS, software version, firewalls, and policies but also the physical 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)

What we have at Edinboro

# Cloud Hosting

Ignore the man behind the curtain

**Cloud hosting** is the newest buzzword in shared hosting services.

The advantages are

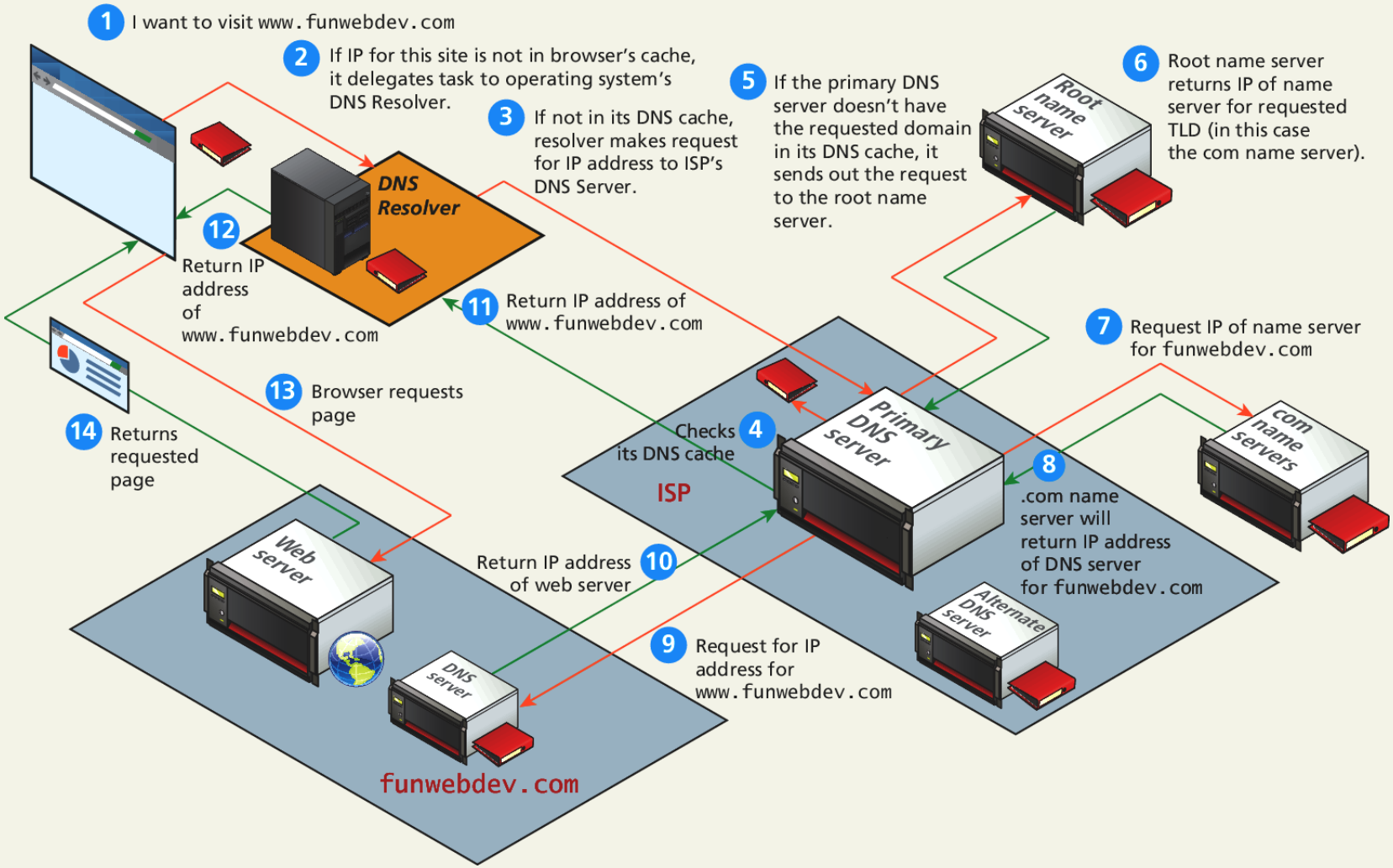
- scalability, where more computing and data storage are needed and
- The redundancy of a distributed solution
- Resources such as disk space come from a network of servers, rendered as needed.
- Manage bursts of requests easily
- Google
- Amazon Web Services

Section 2 of 5

# DOMAIN AND NAME SERVER ADMINISTRATION

# Domain Name System

Better than remembering IP addresses



# 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.

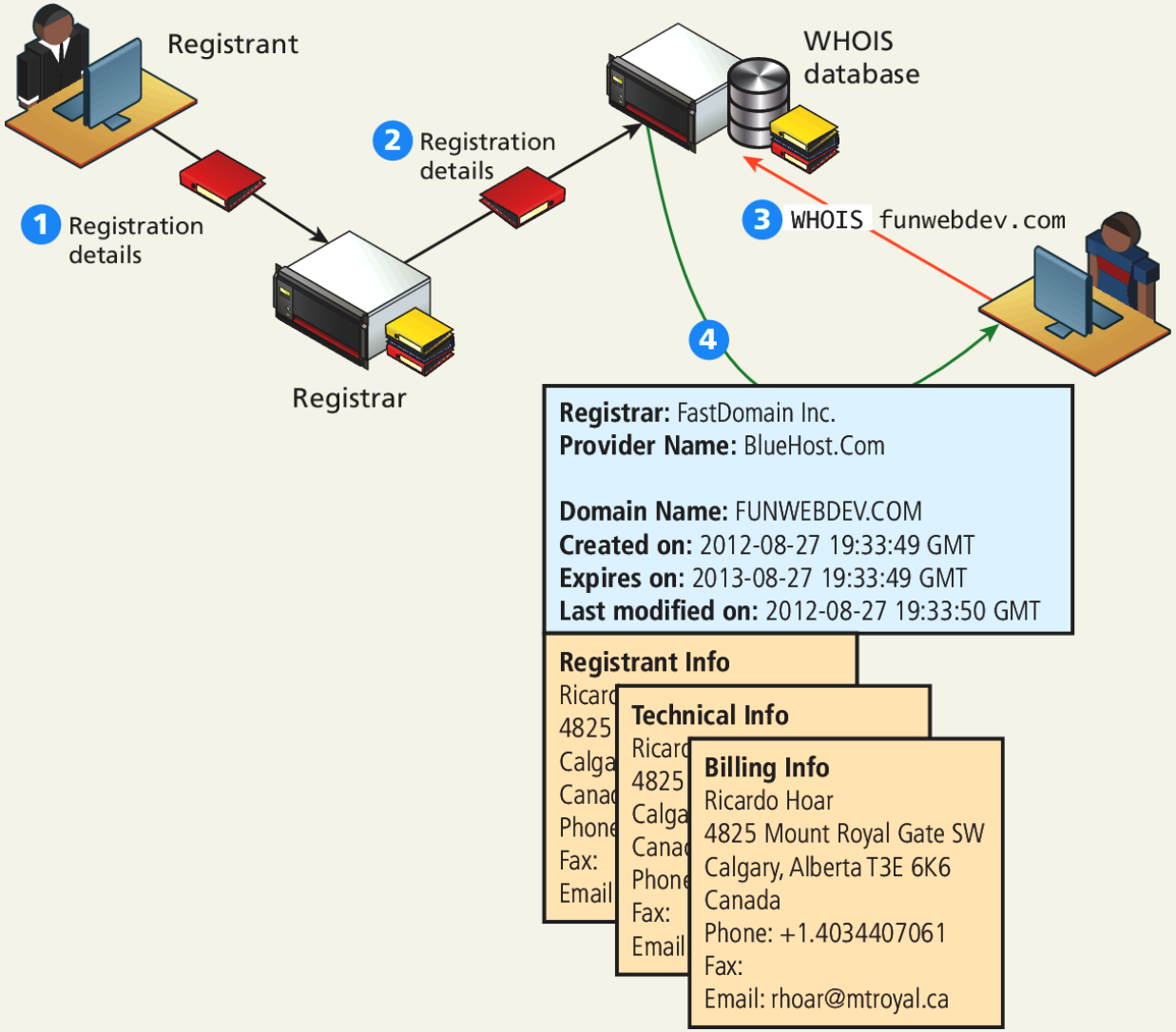
# Registering a Domain Name

## 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.
- I think we need to go to a web site to run whois:
- http://whois.icann.org/

# Whois

## A Visualization



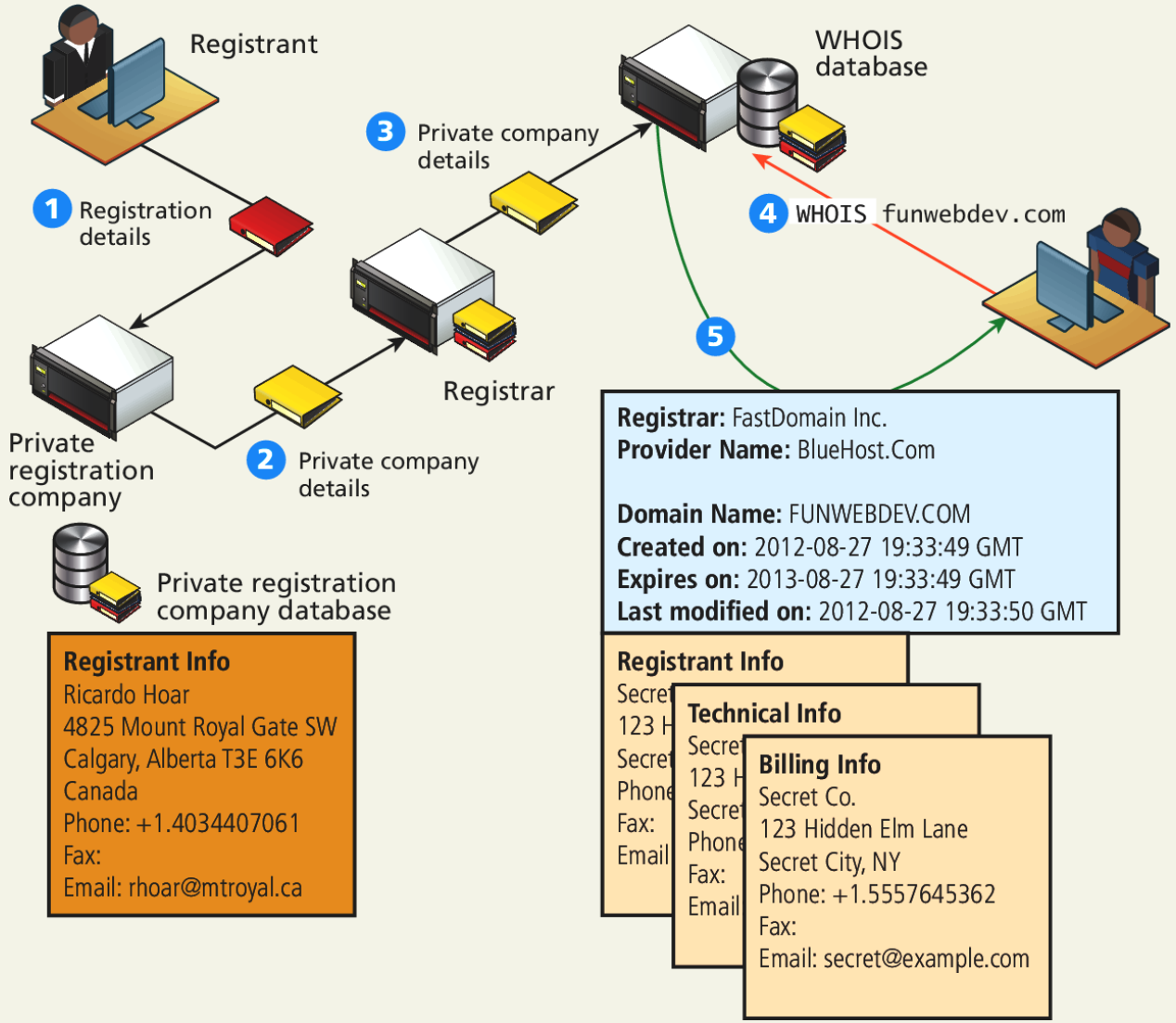
# Whois

## 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



# 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

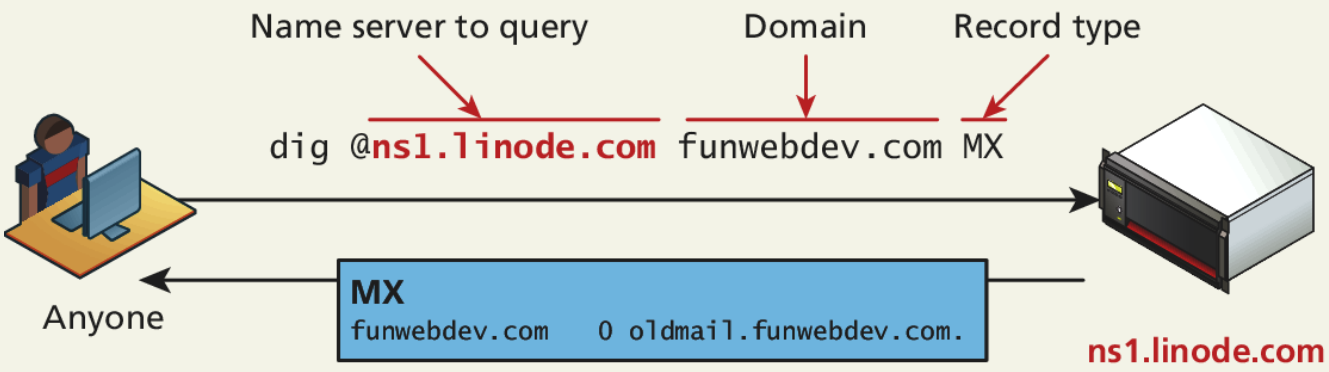
- Updating records in DNS may require at least 48 hours to ensure that the changes have propagated throughout the system.
- After updating your name servers with the registrar, it's a good practice to "dig" on your TLD servers to confirm that the changes have been made.
- Dig is a command that lets you ask a particular name server about records of a particular type for any domain.

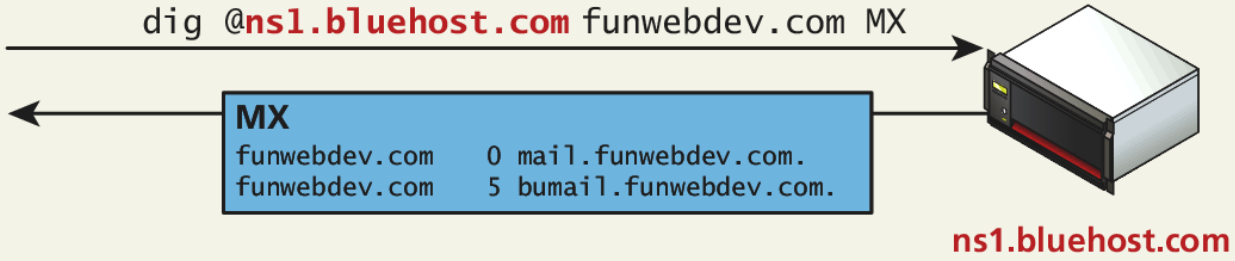
# Checking Name Servers

Dig it

We can try:
http://www.digwebinterface.com/

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



# 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** 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

## 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

# Reverse DNS

in-addr.apra

http://mxtoolbox.com/ReverseLookup.aspx

**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**

147.64.234.215 becomes the PTR entry

PTR 147.64.234.215 visit.edinboro.edu

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.

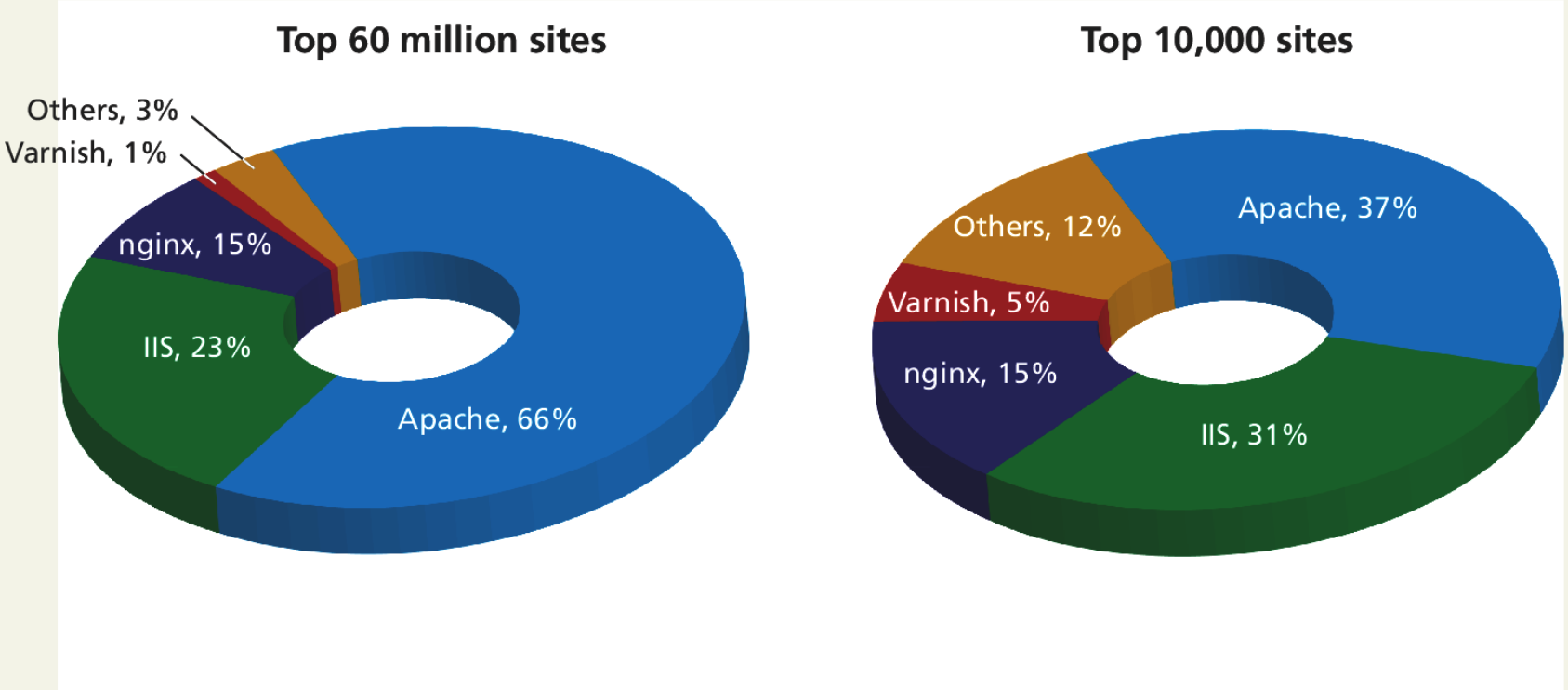
Section 3 of 5

# 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.



# 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**

# 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

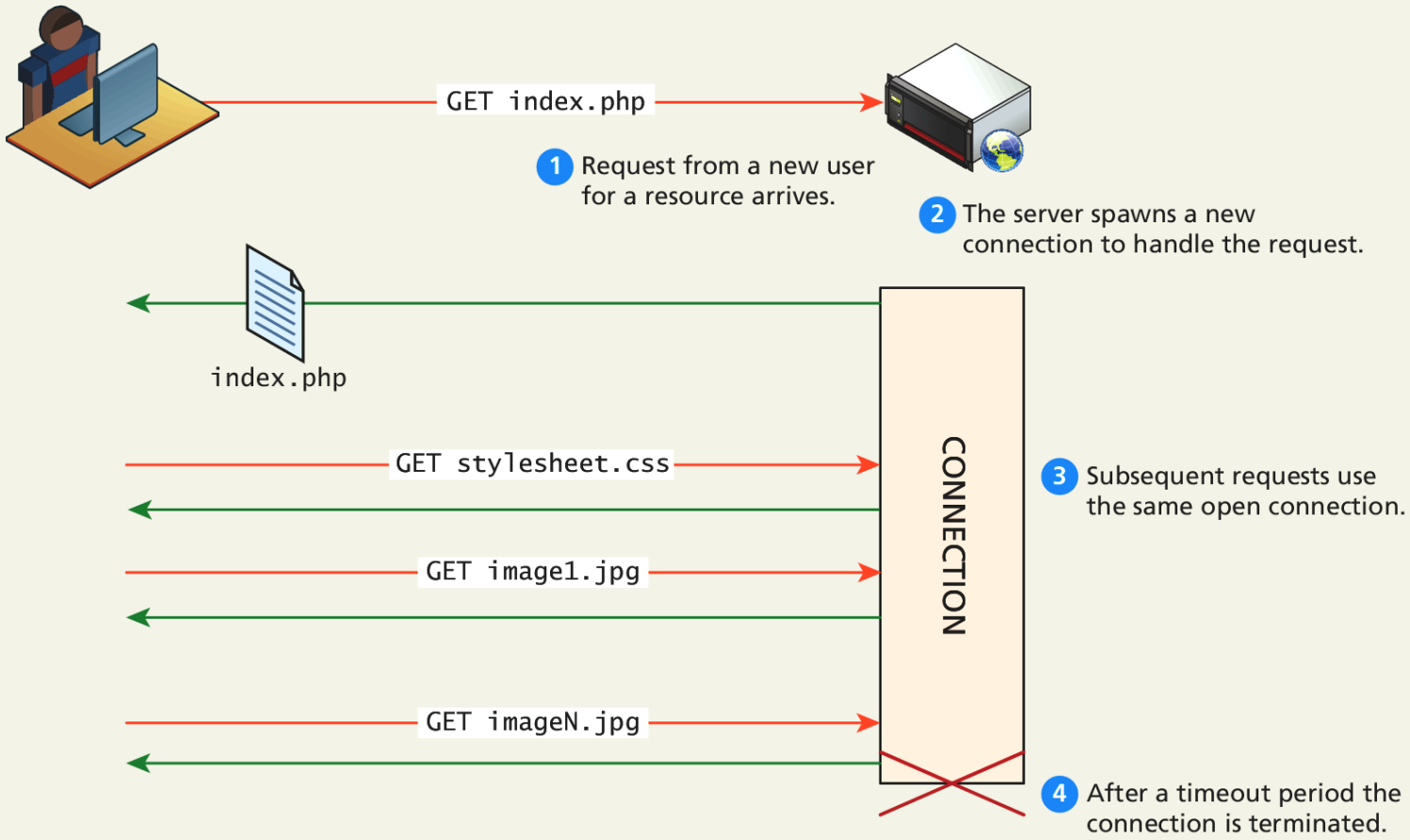
and GUI (Windows)

6. Reboot

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**

# Connection Management

Fine tuning your server



# 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

# File Ownership and Permissions

A review for many

Apache runs as its own user (sometimes called Apache, WWW, or HTTP depending on configuration). 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

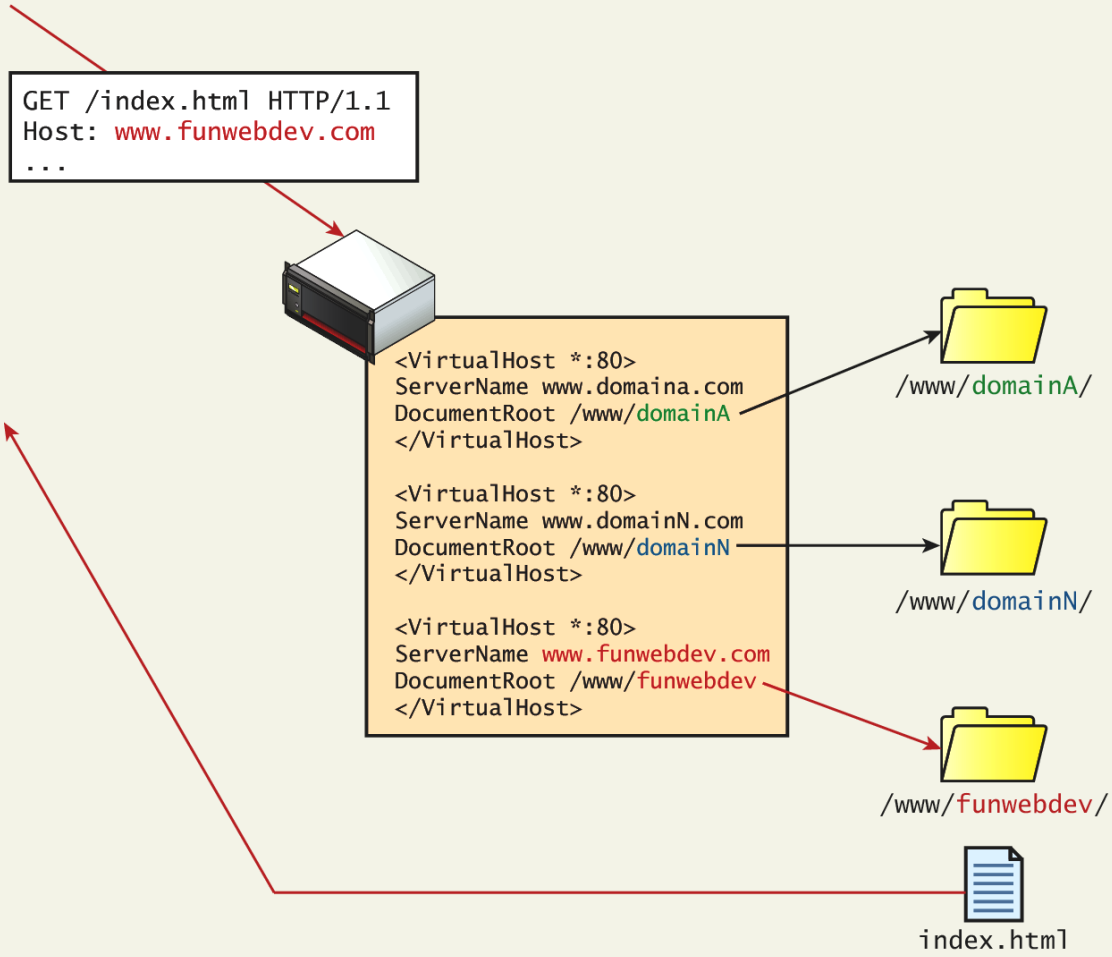
- which IP and port to listen on
- what file system location to use as the root for that domain.
- NameVirtualHost allows you to use domain names instead of IP addresses. This means many domains on 1 IP address!

```
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



# Handling Directory Requests

## The index files

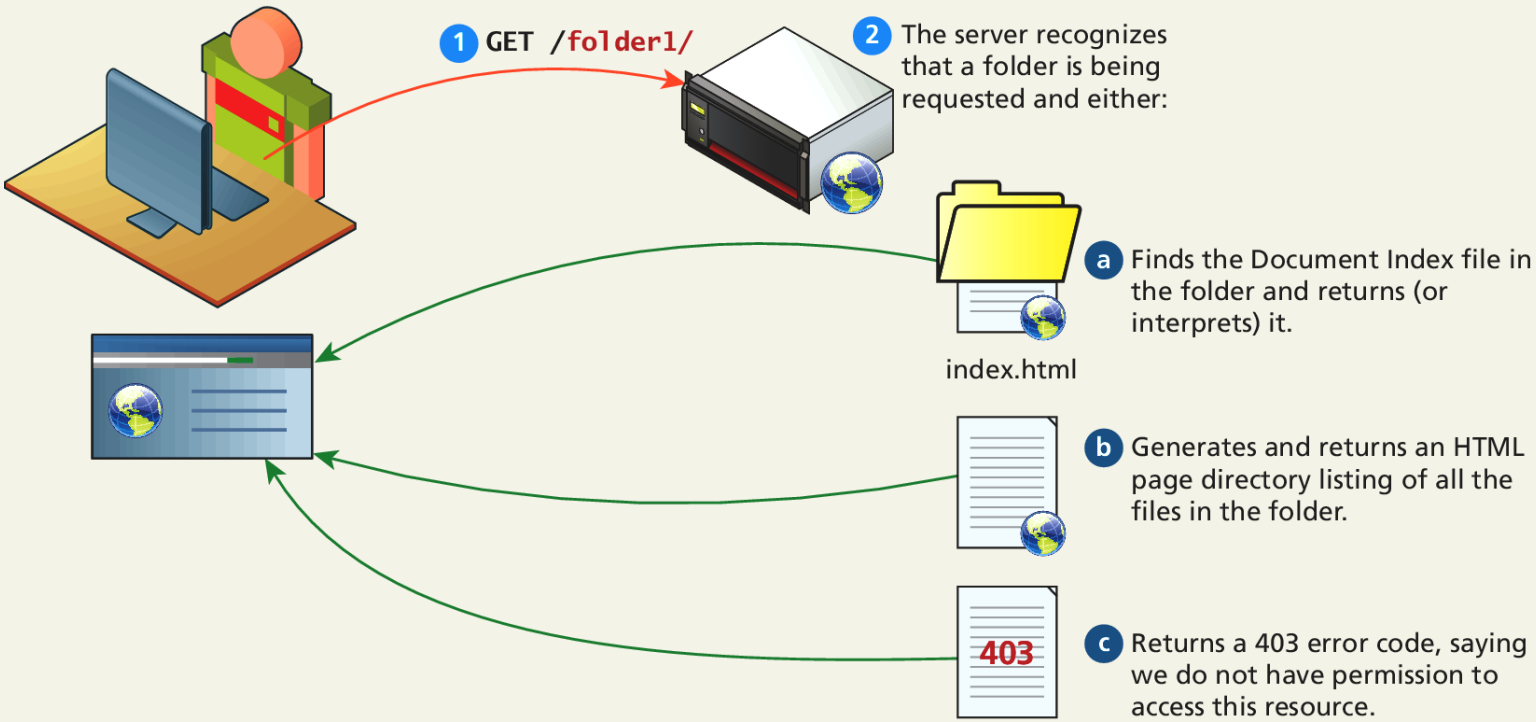
In practice, users normally request a domain's homepage URL without specifying what file they want.

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?




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

# 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.

Mapped the request to a particular file location using the connection management options

The server sends the requested file, along with the relevant HTTP headers

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

# Responding to File Requests

Which files get interpreted

http://www.freeformatter.com/mime-types-list.html

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

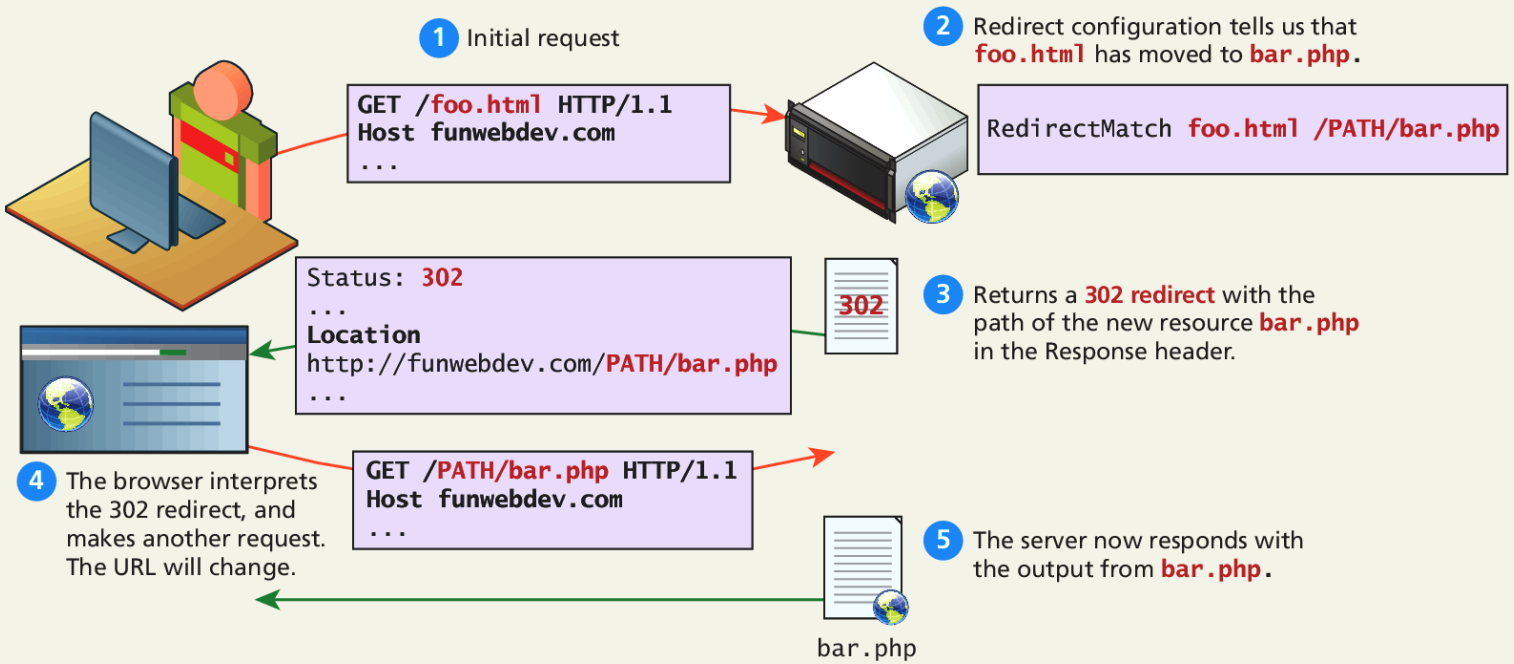
It is a better practice to inform users that their old pages have moved, using a HTTP **302** header

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

# Public Redirection

Two requests required, and everybody knows

– this is the HTTP 302 response



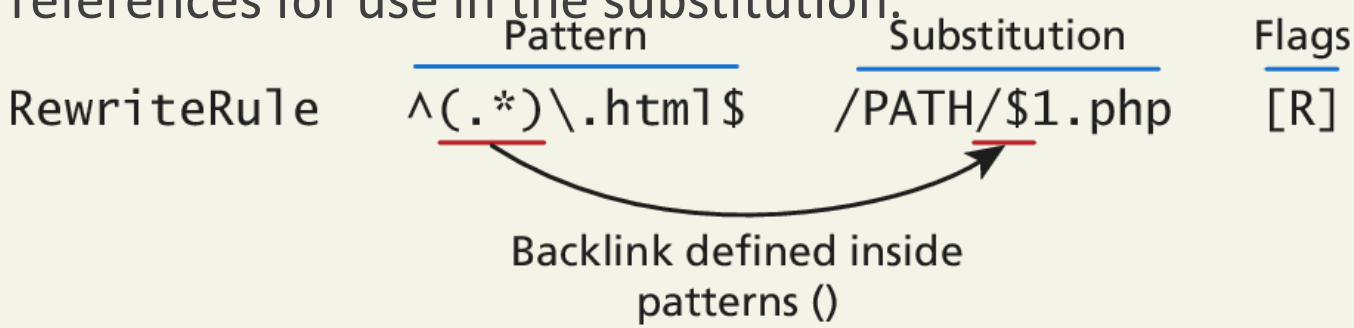
# Public Redirection

## The RedirectRule Directive

You can make a RewriteRule directive

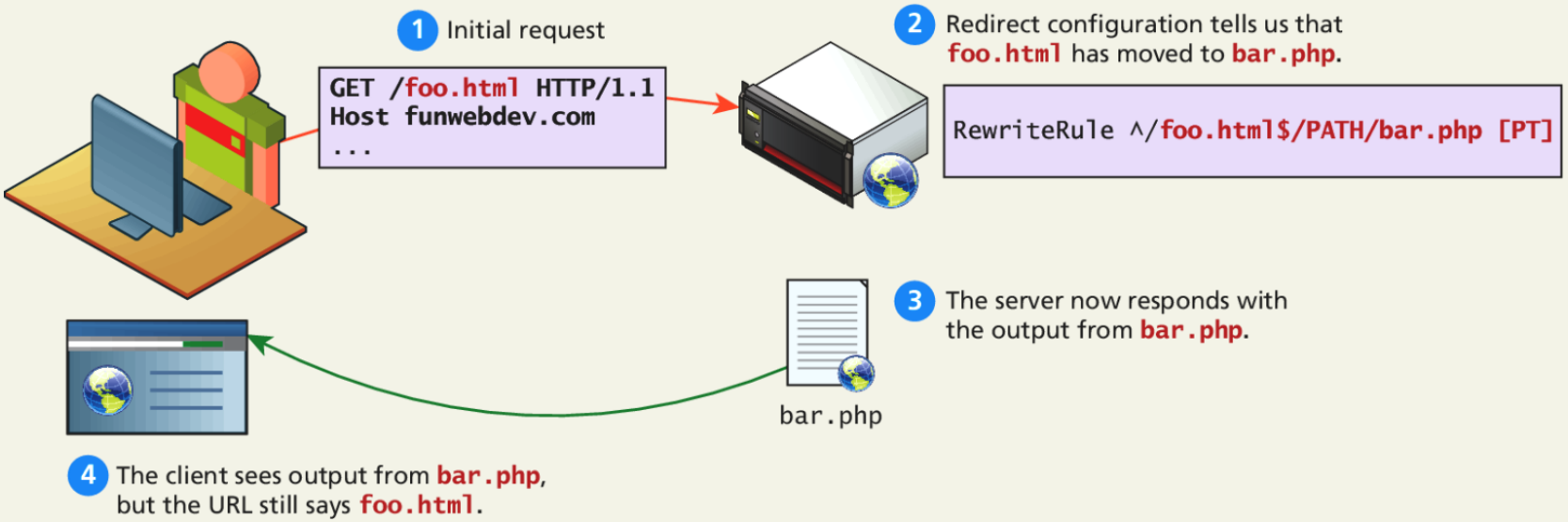
- consists of three parts:
  - the pattern to match,
  - the substitution, and
  - Flags

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



# Internal Redirection

One fewer requests



# 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]**

http://regexlib.com/CheatSheet.aspx?AspxAutoDetectCookieSupport=1

# 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.

| | Test string | Condition | (Optional) Flags |
|---|---|---|---|
| RewriteCond | %{REMOTE_ADDR} | ^192\.168\. | |

# Conditional ReWriting

An advanced example

Linking to image on another site

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.

# 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

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.

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.

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

## 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.

```
/var/log/linuxserver/linux.log {
rotate 7
daily
compress
delaycompress
missingok
notifempty
create 660 linuxuser linuxuser }
```

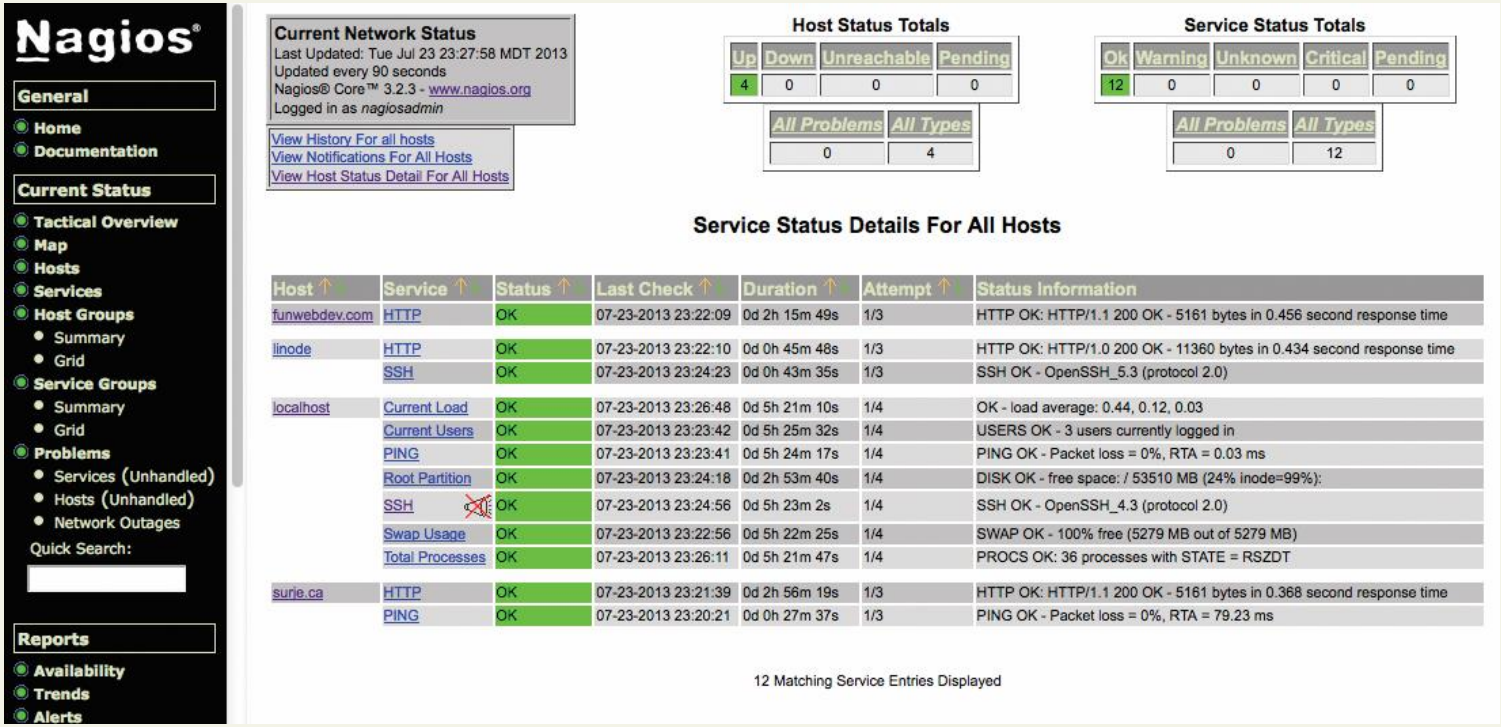
7 rotations
Run daily
Compress
Exclude yesterday and empty
660 permissions
Can email logs also

# External Monitoring

## Test the network

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

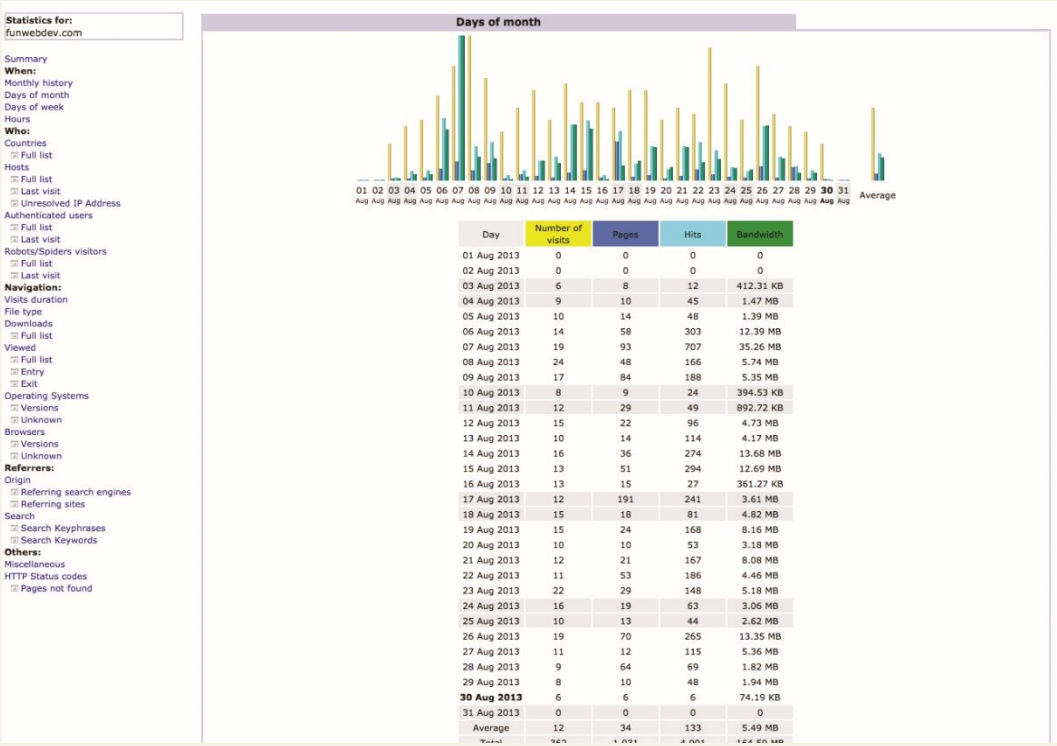
Much like internal logs, external monitoring logs can be used to generate uptime reports and other visual summaries of your server.



# 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 |

# 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

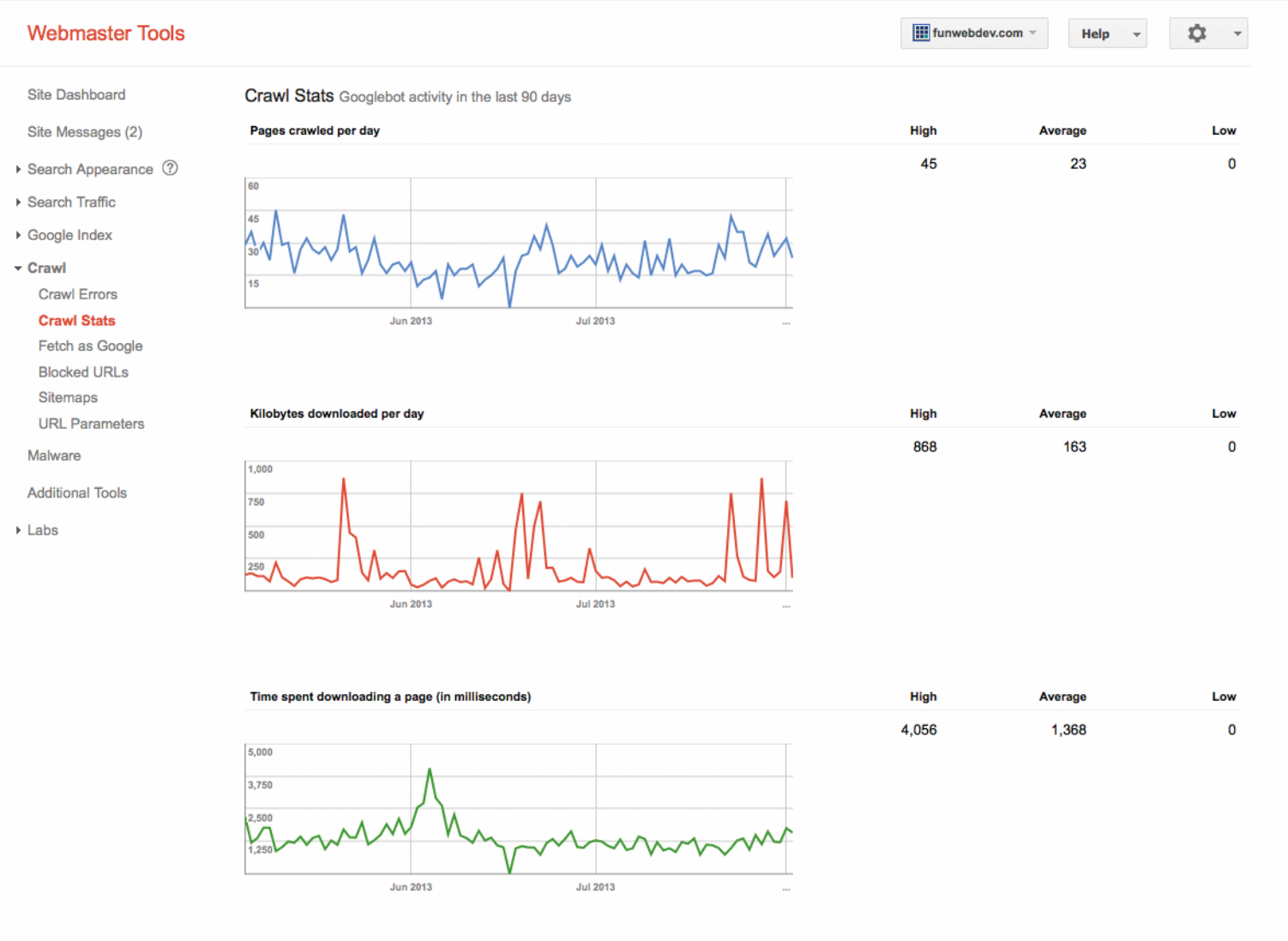
These tools provide information about

- 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



# 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