

GNU/Linux Servers

Prasad

Contents

- Apache HTTP Server
- Squid Proxy and Cache
- Samba Server
- FTP Server

Apache Server

Introduction to HTTP

- HTTP
 - Hyper Text Transfer Protocol
 - Plain text data transfer
 - rfc1945 and rfc2616
- HTTPS
 - Secure version of HTTP
 - Encrypted
 - rfc2818
- <http://www.ietf.org/rfc/rfc<number>.txt>

Introduction to Apache

- The Apache group was formed in 1995
- Apache => A Patchy Server
 - Largely because the patches to NCSA server that resulted in this new webserver. It actually got its name from the native American Tribe
- Apache Version 1.0 was released in December, 1995
- World's most used web-server since April, 1996
 - http://news.netcraft.com/archives/web_server_survey.html

Scope

- Single Website and Virtual Hosting
- Access Configuration
- Security
 - mod_ssl
 - Other security related configurations
 - Digital certificates from OpenSSL for HTTPS
- Doing it on a RedHat based distribution

Configuration Contexts

- Server level configuration
- Virtual Host level configuration
 - one.example.com and two.example.com could be hosted on the same server using name based virtual hosts.
 - IP based virtual hosting, Port based virtual hosting
- Directory/Location level configuration
- Local Configuration
 - Done using “.htaccess” files in a per-directory basis

Core Directives

- Server Parameters
 - Logging Information
 - Eg: LogFormat
 - User, Group
 - Connection Parameters
 - Eg: TimeOut and KeepAlive
 - Server Information
 - Eg: ServerRoot, ServerTokens
- <http://httpd.apache.org/docs/2.2/mod/core.html>

Defining a Server

- DocumentRoot
 - Root of all the HTML documents for this server
- Alias, ScriptAlias
- Directory definitions
- Other Options

Directory Definitions

- Access Permissions
 - Allow, Deny
 - AuthType, AuthName, AuthUserFile, Require, AuthGroupFile
- Options [+|-]option [+|-]option ...
 - All, ExecCGI, FollowSymLinks, Includes, Indexes, SymLinksIfOwnerMatch and MultiViews
- AllowOverride
 - Options that can be overwritten in per-directory configurations

Single Host / Multiple Hosts

- Single Host
 - The host definitions go into the main configuration
 - Any connection to the server returns the same set of pages
- Multiple Hosts called Virtual Hosts
 - Name based VirtualHost
 - Multiple domain names to the same IP address
 - IP based VirtualHost
 - Multiple IP addresses to the same machine
 - Combinations of Name and IP based virtual hosting is possible

Typical Configurations

- Single Host
 - RedHat based distributions will see that everything is put in the main server without using the VirtualHost directive
 - Debian based distributions generally have a virtual host defined. When there is only one virtual host defined, it is the default as long as it meets IP address criteria
- Virtual Hosting
 - Examples follow
- Lets take a look at the configuration file before proceeding

Name Based Virtual Hosting

```
NameVirtualHost *:80
<VirtualHost *:80>
    ServerName one.example.com
    DocumentRoot /var/www/one.example.com
    ...
    ServerAdmin one@example.com
</VirtualHost>
```

- Name virtual host on port 80
- All requests to one.example.com are handled by this virtual host. If no other virtual host exists, then all the requests are handled by this vhost itself.
- DEMO of NAME BASED VIRTUAL HOSTING

IP Based Virtual Hosting

```
Listen 192.168.36.1:80
Listen 172.19.58.23:80
<VirtualHost 172.19.58.23:80>
    ...
</VirtualHost>
<VirtualHost 192.168.36.1:80>
    ...
</VirtualHost>
```

- Listens on port 80 of two different IP address
- Each IP address is handled by different virtual host

Virtual Host Combinations

- NameVirtualHost can be used with IP addresses instead of using a wildcard with the port number
- More than one NameVirtualHost can be used
- one.example.com when referred by a client from 172.19.0.0 network can be served by one host and the same one.example.com when referred to by a 192.168.30.0 network could be served by a different host!

Access Configuration

- Allow from
 - Allow access from the given networks and addresses or when the environment satisfies a condition
- Deny from
 - Deny access from the given networks and addresses or when the environment satisfies a condition
- What if you want to deny 192.168.36.0/24 but allow 192.168.36.20?

Access Configuration

- Order
 - Control the order of evaluation of Deny and Allow directives
 - Deny, Allow
 - Evaluate Deny before Allow
 - Access allowed by default
 - Anything that is not in Deny or is in Allow will be allowed
 - Allow, Deny
 - Evaluate Allow before Deny
 - Access denied by default
 - Anything that is not in Allow or is in Deny is denied

Access Configuration Example

```
<Directory ...>  
    Order Deny, Allow  
    Deny from 192.168.36.0/24  
    Allow from 192.168.36.20  
</Directory>
```

- This handles our problem... denies 192.168.36.0/24 but allows 192.168.36.20 and all others

Authentication

- AuthType
 - Authentication Types, Basic or Digest
- AuthName
 - Identify the server to the user for authentication (Realm)
- Require
 - Require user [userid] [userid]
 - Require group [groupname] [groupname]
 - Require valid-user

Authentication

- Authentication Provider
 - AuthUserFile and AuthGroupFile
 - AuthDBMUserFile and AuthDBMGroupFile
 - LDAP
 - Other providers
- AuthUserFile can be created using 'htpasswd' command for plain authentication and 'htdigest' for digest authentication

SSL Enabling

- Changing the certificate path and the keypath would do the required in most cases
- Create the certificates
 - `openssl genrsa -des3 -out server.key 1024`
 - `openssl rsa -in server.key -out server.pem`
 - `openssl req -new -key server.key -out server.csr`
 - `openssl x509 -req -days 365 -in server.csr -signkey server.key -out server.crt`
- Set `SSLCertificateFile` and `SSLCertificateKeyFile`

Apache can do more...

- mod_proxy can take care of proxy, both reverse and forward
- mod_security can scan the incoming requests for attacks and can proactively prevent attacks on the server
- mod_userdir enables “~username” urls for users
- Perl, Python and Ruby have script engines that come as modules to Apache
- Apache is the most used webserver, since 1996 :)

Squid Proxy and Cache

Introduction to Proxy

- In a typical setup a Web-Proxy requests pages from the Internet on behalf of the clients on the local network and serves them to the local clients.
 - Enhances security on LAN
 - When caching is enabled, gives a better browsing experience
- Proxy servers also cache data and avoid redundant and repeated requests to servers for the same data.

Introduction to Squid

- Web Proxy and Cache for HTTP, FTP
- Cache Hierarchies
- DNS Lookup Cache
- Reverse Proxy to accelerate speed of web servers
- Access Control Lists
- Bandwidth management

Scope

- Single level Web Proxy Server and Cache setup
- Enabling and disabling different protocols
- Access Control Lists
- Bandwidth management using DelayPools
- Doing it all on RedHat based distributions

Minimal Configuration

- Squid had intelligent defaults for every option
- Change the `cache_dir`
 - A different disk or at least a different partition is recommended
 - Eg: `cache_dir aufs /cache 2000 32 256`
 - aufs filesystem, 2000 MB, 32 Level1 and 256 Level2 directories
- Change the ACLs to allow more computer/people
 - Default allows none

Access Control Lists

- Used to control access to various resources of Squid
- acl name type string
 - acl ournet src 192.168.36.0/24
- acl name type “file”
 - acl badsites dstdomain “/etc/squid/badsites.txt”
- Many options
 - acl can be based on src, dst, dstdomain, srcdomain, srcdom_regex, dstdom_regex, time, port, protocol, method, usernames and many more (see comments in squid.conf)

Access Control Lists

- `http_access [allow|deny] aclname aclname ...`
 - it's an 'AND' of the given aclnames
- `http_access` statements are evaluated sequentially till one of them matches
- `http_access [allow|deny] CONNECT aclname ...`
 - Allows or denies tunnelling to the given ACL
 - SSL and many such protocols that Squid cannot understand are handled in this way
- `http_reply_access` – complimentary to `http_access`

ACL Example

```
acl myusers proxy_auth REQUIRED
acl office_timings M T W H F 09:00-18:00
acl special_users proxy_auth prasad
acl mylan src 192.168.36.0/24
acl restrict_sites dstdomain "/etc/squid/restricted.txt"
http_access allow special_users
http_access deny office_timings restrict_sites
http_access allow myusers mylan
http_access deny all
```

Bandwith Management

- Delay Pools
 - Nothing but bandwidth restrictions
- Three Classes
 - Class 1: Single aggregate bucket
 - Class 2: Aggregate + Bucket for each host in class C
 - Class 3: Aggregate + Bucket for each network in class B

Bandwidth Management - Example

```
delay_pools 2      # 2 delay pools
delay_class 1 2    # Pool1 is a class 2 pool
delay_class 2 3    # Pool2 is a class 3 pool
delay_access 1 allow acl_for_pool_one
delay_access 1 deny all
delay_access 2 allow acl_for_pool_two
delay_access 2 deny all
delay_parameters 1 8000/32000 1000/4000
delay_parameters 2 10000/50000 8000/32000 1000/4000
```

Squid has more...

- Reverse proxy to speed up your web-server
- HTTPS accelerator
- Cache peers and cache hierarchies

Samba Server

Introduction to SMB

- NetBIOS by IBM and Sytec
- NetBIOS + Disk I/O redirection => SMB
 - Server Message Block Protocol by Microsoft
 - Now called the CIFS
 - Common Internet File System
- Windows machines advertise their services and presence on the network using this protocol
- The “Network Neighbourhood”

Introduction to Samba

- Andrew Tridgell published his code in early 1992
- Actual development started two years later
- Opening windows to the wider world
- Samba runs on unix platforms, but speaks to Windows clients like a native windows machine
- Lets you share files and printers over the network
- Works with SMB as well as its latest form CIFS

Scope

- File and Printer sharing
- Access controls
- Doing it all on RedHat based distributions

Sections

- Global Parameters
 - workgroup, server string, load printers, printers, encrypt passwords, encrypt passwords create mask, interfaces, hosts allow etc;
- Share Parameters
 - path, public, only guest, guest ok, writable, read only

Simple smb.conf

```
[global]
workgroup = Work
server string = Prasad
encrypt passwords = yes

[homes]
    comment = Home Directories
    browseable = no
    writable = yes
```

Note: refer to the default `/etc/samba/smb.conf` of redhat for more global options.

Note: The passwords for samba are manipulated using the command `smbpasswd`

Simple smb.conf

```
[printers]
```

```
path = /var/spool/samba
```

```
browseable = no
```

```
printable = yes
```

```
writable = yes
```

```
guest ok = yes
```

```
[everyone]
```

```
path = /var/everyone
```

```
public = yes
```

```
only guest = yes
```

```
writable = yes
```

```
read only = no
```

Samba can...

- Be part of a Windows domain
- It can be the Primary Domain Controller
 - Can provide domain logon
 - domain logons = yes
 - Export home directories to Windows
 - logon drive = d:
 - logon home = `\\server\%u`
 - Needs a compulsory share [netlogon]

FTP Server (vsftpd)

Introduction to FTP

- File Transfer Protocol
 - rfc0959, <http://www.ietf.org/rfc/rfc0959.txt>
- Criticisms
 - Passwords are sent in clear-text
 - Multiple TCP/IP connections needed
 - No integrity check in case of connection failures
- Alternatives
 - SFTP and FTPS for secure copying
 - Secure Copy or SCP is now-a-days largely used

Scope

- A simple FTP server
- Anonymous access
- Security considerations
 - chroot jail
- Doing it all on RedHat based distributions

Minimal Configuration

```
background=YES
```

```
listen=YES
```

- Starting from an empty configuration file, the above lines make the vsftpd server to start and listen for incoming connections
- So the vsftpd is now listening!

Minimal Configuration

```
local_enable=YES
```

- Enable local usernames to login
 - `userlist_enable` permits disabling certain users from logging in
 - FTP is a very insecure protocol, hence all these restrictions

Minimal Configuration

```
anonymous_enable=YES
```

- Enables anonymous FTP

- anonymous ftp will also involve taking care of directories for uploading of files by the anonymous user if uploading files is allowed

Adding a little Security

```
anonymous_enable=YES  
chroot_list_enable=YES  
chroot_local_user=YES  
xferlog_enable=YES  
xferlog_file="/var/log/ftp.log"
```

- Creates a chroot jail when anyone logs-in unless the username is listed in `chroot_local_user`, which is there at `/etc/vsftpd.chroot.list`
- Done, lets try it on the system now!

Thanks!

Prasad <s.prasad@gmail.com>