Web and Proxy Server Configuration

Hiranya Prasad Bastakoti

Contents

Web Server

- A web server is a crucial component of the internet infrastructure that handles client requests and serves website content.
- It combines both software and hardware elements to facilitate the exchange of data between clients and servers.
- At its core, a web server utilizes the HTTP protocol to communicate with clients, which are typically web browsers.
- When a client requests a webpage by entering a URL or clicking a link, the web server receives the request and processes it to deliver the appropriate content back to the client.
- The hardware of a web server consists of physical machines or servers that are connected to the internet.
- These servers are designed to handle and process a large volume of requests simultaneously. They store website files, databases, and other resources required to serve webpages.

- On the software side, web servers employ specialized software applications that handle the processing and delivery of web content. Apache HTTP Server and Nginx are two popular examples of web server software.
- In addition to HTTP, web servers can also support other protocols like SMTP and FTP. SMTP allows the server to handle email-related tasks such as sending and receiving emails, while FTP enables file transfer and storage functionalities.
- Web servers are integral to web hosting services, where they store and serve website data and applications. Web hosting providers offer storage space and resources on their web servers to make websites accessible on the internet.

Common Web Server

- **Apache HTTP Server.** Developed by Apache Software Foundation, it is a free and open source web server for Windows, Mac OS X, Unix, Linux, Solaris and other operating systems.
- **Microsoft Internet Information Services (IIS).** Developed by Microsoft for Microsoft platforms; it is not open sourced, but widely used.
- **Nginx.** A popular open source web server for administrators because of its light resource utilization and scalability. It can handle many concurrent sessions due to its event-driven architecture. Nginx also can be used as a proxy server and load balancer.

- **Lighttpd.** A free web server that comes with the FreeBSD operating system. It is seen as fast and secure, while consuming less CPU power.
- **Sun Java System Web Server.** A free web server from Sun Microsystems that can run on Windows, Linux and Unix. It is well-equipped to handle medium to large websites

Apache Web Server

- Apache Web Server is an open-source web server software that is used to host websites and web applications.
- Apache Web Server is known for its reliability, flexibility, and scalability, making it an ideal choice for businesses of all sizes.
- It is also highly secure, with a range of features designed to protect website from malicious attacks.

Features

- Highly Reliable: Apache is known for its robustness and stability, making it a reliable choice for hosting websites and applications. It has been extensively tested and refined over the years, making it a trusted and dependable server solution.
- Highly Secure: Apache offers a wide range of security features to protect websites and applications from various threats. It supports SSL/TLS encryption for secure data transmission, provides access control mechanisms, and offers modules for handling authentication and authorization.
- Highly Scalable: Apache is designed to handle high-traffic websites and can efficiently manage a large number of concurrent connections. It allows for easy scaling by supporting load balancing techniques, such as mod_proxy_balancer, which distribute the workload across multiple servers.

- Highly Compatible: Apache is compatible with different operating systems, including Unix-like systems (such as Linux, BSD, and macOS) and Windows. It supports multiple protocols, including HTTP, HTTPS, and FTP, making it compatible with a wide range of clients and applications.
- Highly Customizable: Apache allows extensive customization through its configuration files. Administrators can tailor the server's behavior by modifying settings, enabling or disabling modules, and defining rules for handling requests. This flexibility makes it suitable for various use cases and environments.
- Highly Automated: Apache provides automation features that simplify server management and administration. It offers tools like Apache's HTTP Server Control Interface (mod_status), which provides real-time monitoring and server statistics. Additionally, Apache can be integrated with various management and deployment tools, enabling automated configurations and updates.

Configuration of HTTP Apache Server

- **Step 1: Update the server**
- To run updates on the CentOS server, enter the following command:

  $ sudo yum update
- This command will ensure all software is up to date and using the latest versions. The process should take a couple of minutes

- **Step 2: Install a basic firewall**
- This step allows for Apache to run traffic through the Internet. Enter the command below to **install** the firewall.

```
$ sudo yum install firewalld
```

- After entering this command and pressing “Enter”, the user will be prompted for a password and permission to install necessary packages.

- After the installation is finished, the screen will read “Complete!” Next, run the command to **enable** the firewall:

```
$ sudo systemctl enable firewalld
```

- **Start** the firewall by running this command:

```
$ sudo systemctl start firewalld
```

- To ensure the firewall is running properly, complete a status check with this command:

```
$ sudo systemctl status firewalld
```

- The output should read that the firewall is “**active (running)**”.

- **Step 3: Install Apache and configure the firewall**
- First, enter the following command to install the web server:

```
$ sudo yum install httpd
```

- Once complete, configure the default port for **HTTP (80)** and **HTTPS (443)** to run through the firewall using the following commands:

```
$ sudo firewall-cmd --permanent --add-service=http
$ sudo firewall-cmd --permanent --add-service=https
```

Press “Enter” after each entry should return “**success**”. In order for the changes to take effect, the user must reload with this command:

-
  ```
  $ sudo firewall-cmd --reload
  ```

- **Step 4: Confirm Apache is running**
- Once receiving "**success**" after reloading the firewall, the following commands will start Apache and confirm its "**active (running)**" status:

```
$ sudo systemctl start httpd
$ sudo systemctl status httpd
```

- **Step 5: Confirm Apache is working properly**
- To test the connectivity of the Apache server, locate the user’s public IP address by using this command:

```
$ sudo curl ifconfig.me
```

Troubleshooting

Check Apache Configuration

- The first step in troubleshooting Apache web server in Linux is to check the configuration.
- Apache configuration files are located in the /etc/apache2 directory.
- The main configuration file is httpd.conf. This file contains all the settings for the Apache web server. It is important to check this file for any errors or typos. If any errors are found, they should be corrected before proceeding.
- Additionally, it is important to check the other configuration files in the /etc/apache2 directory for any errors or typos.

Check Log Files

- The next step in troubleshooting Apache web server in Linux is to check the log files. Apache log files are located in the /var/log/apache2 directory.
- The main log file is access.log.
- This file contains all the requests made to the Apache web server. It is important to check this file for any errors or suspicious activity. Additionally, it is important to check the other log files in the /var/log/apache2 directory for any errors or suspicious activity.
- **Check Apache Processes**
- The third step in troubleshooting Apache web server in Linux is to check the Apache processes. Apache processes can be viewed using the ps command. This command will list all the Apache processes that are currently running.
- It is important to check these processes for any errors or suspicious activity. Additionally, it is important to check the other processes in the system for any errors or suspicious activity.

- **Check Network Connections**
- The fourth step in troubleshooting Apache web server in Linux is to check the network connections. Network connections can be viewed using the netstat command. This command will list all the network connections that are currently active.
- It is important to check these connections for any errors or suspicious activity. Additionally, it is important to check the other network connections in the system for any errors or suspicious activity.
- **Check Firewall Rules**
- The next step in troubleshooting Apache web server in Linux is to check the firewall rules. Firewall rules can be viewed using the iptables command.
- This command will list all the firewall rules that are currently active. It is important to check these rules for any errors or suspicious activity. Additionally, it is important to check the other firewall rules in the system for any errors or suspicious activity.
- **Check System Resources**
- The sixth step in troubleshooting Apache web server in Linux is to check the system resources. System resources can be viewed using the top command. This command will list all the system resources that are currently being used.
- It is important to check these resources for any errors or suspicious activity. Additionally, it is important to check the other system resources in the system for any errors or suspicious activity.
- **Check Apache Modules**
- The seventh step in troubleshooting Apache web server in Linux is to check the Apache modules. Apache modules can be viewed using the apachectl command.
- This command will list all the Apache modules that are currently installed. It is important to check these modules for any errors or suspicious activity. Additionally, it is important to check the other Apache modules in the system for any errors or suspicious activity.

- **Check System Services**
- The eighth step in troubleshooting Apache web server in Linux is to check the system services. System services can be viewed using the service command. This command will list all the system services that are currently running.
- It is important to check these services for any errors or suspicious activity. Additionally, it is important to check the other system services in the system for any errors or suspicious activity.
- **Check System Updates**
- The ninth step in troubleshooting Apache web server in Linux is to check the system updates. System updates can be viewed using the yum command. This command will list all the system updates that are currently available.
- It is important to check these updates for any errors or suspicious activity. Additionally, it is important to check the other system updates in the system for any errors or suspicious activity.
- **Check Apache Security**
- The tenth step in troubleshooting Apache web server in Linux is to check the Apache security. Apache security can be viewed using the apachectl command. This command will list all the Apache security settings that are currently in place.
- It is important to check these settings for any errors or suspicious activity. Additionally, it is important to check the other Apache security settings in the system for any errors or suspicious activity.

Virtual Hosting

- Virtual hosting allows us to h**ost many domains** on one server. A server may have **extensive resources** like **HDD space, CPU, RAM**, and so on.
- Virtual web hosting refers to the process of running multiple "virtual" web servers on a single physical host computer.
- Using this technique, a single computer can host thousands of independent websites.
- Commercial web hosting service providers often use this technique to allow better manageability, efficiency, and scalability of their service infrastructure.
- Virtual hosting is a method that servers such as web servers use to host more than one domain name on the same computer, sometimes on the same IP address.
- we can use the same server resources for different sites.

Note: https://www.golinuxhub.com/2014/08/what-are-different-types-of-virtual-web/

Types of Virtual Hosting

1. **Internet protocol (IP) address**. Use a different IP for each domain, but point them to one server. Allow that one server to resolve multiple IP addresses.

- In this method, each domain is assigned a unique IP address, and all IP addresses are associated with the same physical server.
- When a request is made to access a particular domain, the DNS (Domain Name System) resolves the domain name to its corresponding IP address.
- The request then reaches the server, which uses the IP address to determine which website to serve.
- This method allows for clear separation of websites as each domain has its own IP address

2. Name. Use one IP for all domains on the server. During connection, ask your visitors which site they'd like to visit. After that query, resolve the visit to the proper site.

- With this approach, all domains hosted on the server share the same IP address.
- When a request is made, the server checks the HTTP header sent by the client's browser, which contains the requested domain name.
- Based on the domain name, the server determines which website to serve.
- It uses the information in the HTTP header to route the request to the appropriate virtual host or website.
- This method is known as name-based virtual hosting and is commonly used as it allows hosting multiple domains without the need for additional IP addresses.

3.Port. Assign each website to a different port on the server.

- In this method, each website is assigned a different port on the server.
- A port is a communication endpoint that allows different services or applications to run on the same server.
- By assigning each website to a specific port, the server can distinguish which website the incoming request is intended for.
- For example, Website A could be assigned port 80, Website B could be assigned port 81, and so on. However, this approach is less common than the previous two methods, as it may require visitors to specify the port number in the URL when accessing the website, which is not user-friendly.

Advantages

- **Virtual Server Hosting** is simplified and enables saving time, space, and costs.
- Centralized administration and easily compatible with apps.
- Greater accessibility and simple disaster recovery functions.
- The functionalities for controlling backups and can use several OS environments on the same PC.
- Manageable access to **crucial data** and intellectual resources by securing them in the data center.
- Practical use of space allowing greater availability of space in racks as fewer physical devices are installed.
- Transferring servers to new hardware instantly.
- *Configuration steps https://www.cloudpanel.io/tutorial/how-to-host-multiple-websites-on-one-server/*

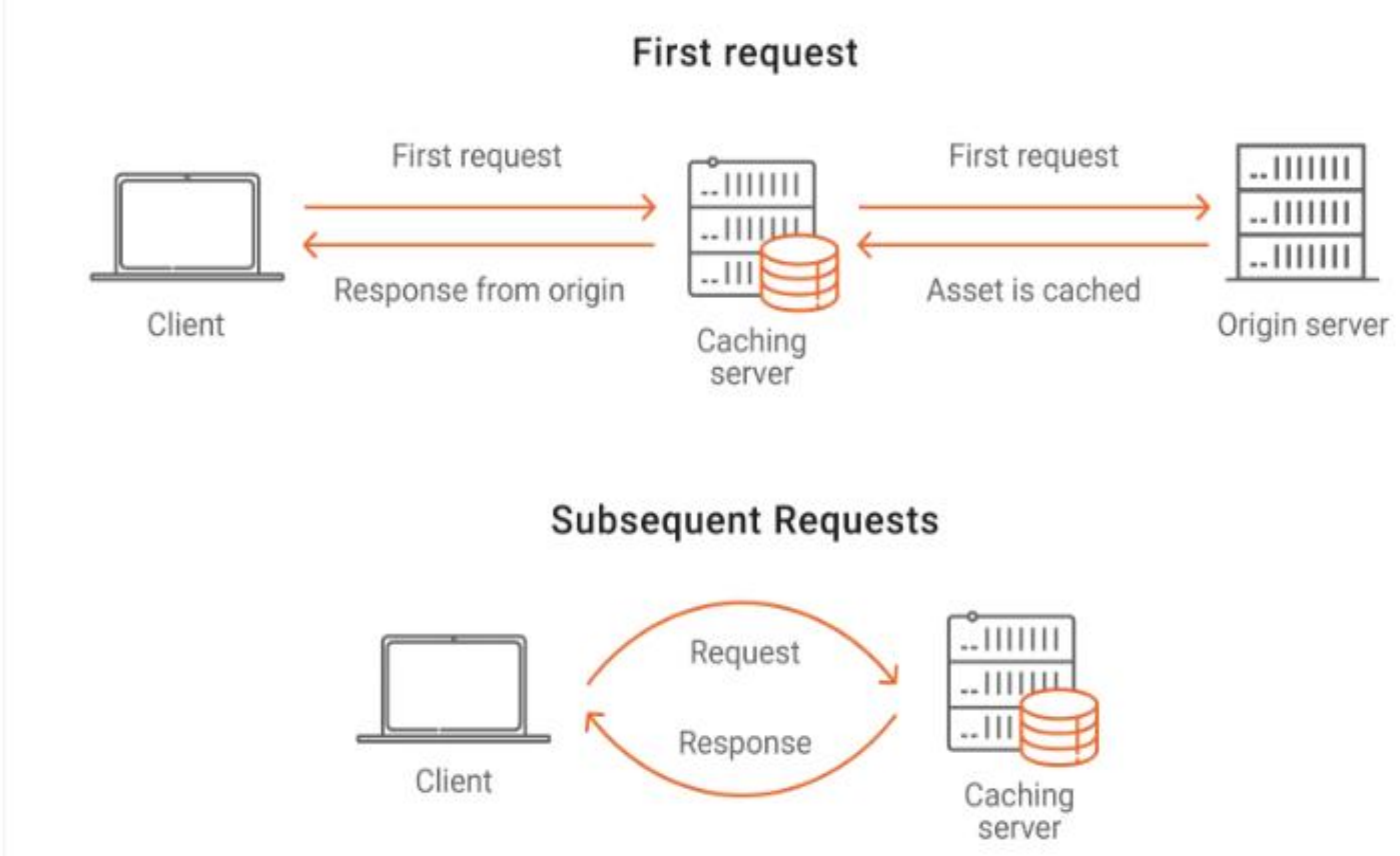
Disadvantages

- The most significant drawback of Virtual Server Hosting is that whenever the server goes offline, all the **hosted websites will also go down** So, to avoid this, companies have to install a cluster of servers.
- The creation of instances, control, configurations, and saving data on virtual machines is very important in virtual environments.
- Difficult to get indirect access to hardware, for instance, particular cards or USB devices.
- For now, there are no consolidated systems to estimate the performance of virtualized environments.
- When multiple virtual machines are operating on the same host, performance may decrease if the PC it is working on lacks enough power.
- High consumption of RAM as every virtual machine will utilize a separate area of the same.

HTTP Caching

- HTTP caching is a mechanism used by web browsers and servers to store and reuse previously accessed resources, such as web pages, images, or scripts.
- Its purpose is to enhance the performance and efficiency of web applications by reducing the amount of data that needs to be transmitted over the network

HTTP Caching



How does HTTP Cache works?

- The website page requests a resource from the origin server.
- The system checks the cache to see if there is already a stored copy of the resource.
- If the resource is cached, the result will be a cache hit response and the resource will be delivered from the cache.
- If the resource is not cached, cache loss will result and the file will be accessed in its original source.
- After the resource is cached, it will continue to be accessed there until it expires or the cache is cleared.

Types of Cache

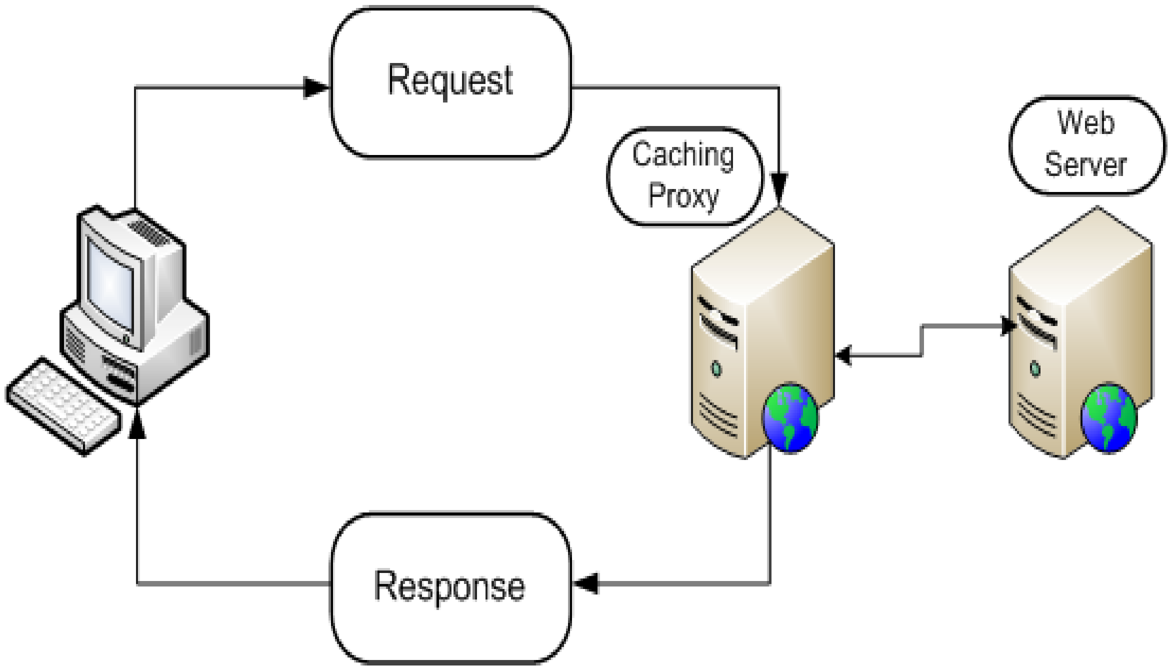
- Browser cache - this storage is done in the browser. All browsers have a local storage, which is usually used to retrieve previously accessed resources. This type of cache is private since stored resources are not shared.
- Proxy cache - this storage, also called intermediate caching, is done on the proxy server, between the client and the origin server. This is a type of shared cache as it's used by multiple clients and is usually maintained by providers.
- .

- Gateway cache - also called reverse proxy, it’s a separate, independent layer, and this storage is between the client and the application. It caches the requests made by the client and sends them to the application and does the same with the responses, sending from the application to the client. If a resource is requested again, the cache returns the response before reaching the application. It’s also a shared cache, but by servers not users.
- Application cache - this storage is done in the application. It allows the developer to specify which files the browser should cache and make them available to users even when they are offline

Proxy Caching Server

- Caching is a feature in which the proxy server saves local copies of the files that clients request so that it can serve them quickly from the cache when they are requested again by the same or other clients.
- A proxy server verifies and forwards incoming client requests to other servers for further communication.
- A proxy server is located between a client and a server where it acts as an intermediary between the two, such as a Web browser and a Web server.
- Proxy caching allows a server to act as an intermediary between a user and a provider of web content.
- When a user accesses a website, proxies interpret and respond to requests on behalf of the original server.
- The proxy server’s most important role is providing security.

- Web cache proxy servers are an essential part of the modern internet.
- They are used to improve the performance of web applications, reduce bandwidth costs, and protect against malicious attacks.
- Web cache proxy servers work by storing copies of web pages and other content on the server, so that when a user requests a page, the server can quickly serve the cached version instead of having to fetch the page from the original source.
- This can significantly reduce the amount of time it takes to load a page, as well as reduce the amount of bandwidth used.


Request
Caching Proxy
Web Server
Response

Benefits

Improved Performance: A cache proxy server stores copies of frequently accessed web pages, images, and other resources. When a user requests a particular resource, the server first checks if it has a cached copy.

- If available, the server delivers the content directly from the cache, eliminating the need to retrieve it from the original source.
- This results in faster response times and improved overall performance, as the content is served locally without the delays associated with network latency and server processing.

Reduced Bandwidth Costs: By caching and serving content locally, a cache proxy server reduces the amount of data that needs to be transferred over the network.

- When multiple users request the same resource, the server can deliver it from the cache to subsequent users, avoiding redundant data transfers from the original source.
- This reduces the bandwidth consumption and can lead to significant cost savings, especially in scenarios where bandwidth is limited or expensive.

Increased Security: Web cache proxy servers can enhance security by acting as a shield between users and the internet. They can be configured to filter and block access to malicious or inappropriate websites, thereby safeguarding users from potential threats.

- By inspecting incoming and outgoing traffic, the proxy server can detect and block suspicious activities, such as malware downloads or unauthorized access attempts.
- Moreover, caching can also provide protection against distributed denial-of-service (DDoS) attacks by absorbing a significant portion of the traffic and reducing the impact on the origin server.

Offline Access and Reliability: Web cache proxy servers offer offline access to previously accessed content. When a user requests a web page or resource that has been cached, they can still access it even if the original source or internet connectivity is temporarily unavailable.

- This is particularly useful in situations where users need to access critical information or resources even in offline or intermittent connectivity environments.
- By serving content from the cache, the proxy server ensures that users can continue accessing certain resources without disruptions.

Optimized Content Delivery: Cache proxy servers can employ techniques such as content compression and content delivery network (CDN) integration to further optimize content delivery. They can compress content before caching it, reducing the file sizes and improving the transfer speeds.

- Additionally, cache proxy servers can integrate with CDNs, which distribute cached content across multiple servers located in different geographical regions.
- This ensures that content is delivered from a server that is closer to the user, minimizing latency and improving the overall user experience.

Offline Accessibility: Cache proxy servers allow for offline access to previously accessed content.

- If a user requests a web page or resource that has been cached, they can still access it even if the original source or internet connectivity is unavailable.
- This is particularly useful in environments with intermittent or unreliable internet connections, as users can continue to access critical resources from the cache without disruption.

Examples

Squid: Squid serves as a popular open-source cache proxy server, supporting various protocols such as HTTP, HTTPS, FTP, and more.

- It provides a range of functionalities, including caching, access control, authentication, and content filtering.
- Squid offers extensive customization options, making it adaptable for diverse network environments.

Nginx(engine x): While primarily recognized as a web server, Nginx also functions as a cache proxy server.

- Leveraging its reverse proxy capabilities and caching modules, Nginx efficiently caches and serves static content.
- This enhances performance and reduces bandwidth consumption.

Varnish: Developed as a high-performance HTTP cache proxy server, Varnish is designed to accelerate content delivery.

- It stores frequently accessed web content in memory and directly delivers it from the cache, resulting in rapid response times. Varnish is commonly deployed in front of web servers to enhance overall performance.

Configuring Squid Cache Server

1. Update Software Repositories

- Open terminal and run the following command to update CentOS software repositories.

```
$ sudo yum -y update
```

2. Install Squid Package

- Run the following command to install Squid Package.

```
$ yum -y install squid
```

Contd..

If you see any prompts during installation, enter Y.

Once Squid server is installed, start it with the following command.

```
$ sudo systemctl start squid
```

If you want to automatically start squid at the time of boot, run the following command.

```
$ sudo systemctl enable squid
```

If you want to view the status of your squid server, run the following command

```
$ sudo systemctl status squid
```

3. Configure Squid Proxy Server

- Next, we will configure Squid Proxy Server. Open its configuration file found at /etc/squid/squid.conf using a text editor.

```
$ sudo vi /etc/squid/squid.conf
```

By default, Squid Proxy server is configured to block all incoming HTTP traffic using the following line.

- `http_access deny all`

Change it to

- `http_access allow all`

Restart Squid proxy to apply changes.

- `$ sudo systemctl restart squid`

4.Create Access Control List (ACL)

- If you are unable to access your proxy server from outside its network, then create an access control list. Open Squid Proxy configuration file
- $ sudo vi /etc/squid/squid.conf
- Add the following line to allow access from 54.43.32.21
- acl localnet src 54.43.32.21
- You can also allow access from an IP range using CIDR notations
- acl localnet src 54.43.32.0/24

6.Set up basic authentication

Squid also supports basic authentication. Run the following command to install httpd-tools

```
yum -y install httpd-tools
```

Once the installation is complete, create new file

```
$ sudo touch /etc/squid/passwd && chown squid /etc/squid/passwd
```

Use the following command to create password, replace user_name with your username

```
htpasswd /etc/squid/passwd user_name
```

Enter the password for this new user, when prompted.

Open Squid Proxy configuration file

```
$ sudo vi /etc/squid/squid.conf
```

Add the following lines.

- auth_param basic program /usr/lib64/squid/basic_ncsa_auth /etc/squid/passwd
- auth_param basic children 5 auth_param basic realm Squid Basic Authentication auth_param basic credentialsttl 2 hours
- acl auth_users proxy_auth REQUIRED http_access allow auth_users

Restart Squid Proxy Server to apply changes.

```
$ sudo systemctl restart squid
```

7.How to Block Websites using Squid Proxy

- Squid proxy also allows you to block specific websites. Create a new file to list blocked websites

```
$ sudo vi /etc/squid/blocked.acl
```

- Add the websites to be blocked, one per line. Each starting with a dot.

```
.twitter.com
.facebook.com
.pinterest.com
```

open Squid configuration file

```
$ sudo vi /etc/squid/squid.conf
```

Add the following lines above your ACL list

```
acl blocked_websites dstdomain “/etc/squid/blocked.acl”
http_access deny blocked_websites
```

Save and close the file. Restart Squid Proxy to apply changes.

```
$ sudo systemctl restart squid
```

Proxy ACL

- An Access Control List (ACL) is a set of rules used to control network traffic and enhance network security by filtering incoming or outgoing packets based on defined criteria.

Key features of ACLs include:

- Rule Matching: ACL rules are sequentially matched, starting from the first rule and progressing through subsequent rules. Once a matching rule is found, further comparisons are halted, and the corresponding action is executed.
- Implicit Denial: At the end of every ACL, there is an implicit denial statement. If no rule matches the packet, it will be discarded.(default answer)
- Inbound Access Lists: When applied to the inbound packets of an interface, the ACL processes the packets before routing them to the outbound interface.
- Outbound Access Lists: When applied to outbound packets of an interface, the packets are first routed and then processed at the outbound interface.

Types of ACLs:

- Standard Access Lists: These ACLs are based on source IP addresses only and permit or deny the entire protocol suite.
- They do not distinguish between specific IP traffic types (e.g., TCP, UDP).
- Standard ACLs use numbers 1-99 or 1300-1999 for identification, with the specified address as the source IP.
- Extended Access Lists: These ACLs utilize source IP, destination IP, source port, and destination port.
- They allow for specifying which IP traffic should be allowed or denied. Extended ACLs use range 100-199 or 2000-2699 for identification.

Advantages of ACLs:

- Improved network performance.
- Enhanced network security by filtering unwanted packets.
- Control over network traffic, allowing for permitting or denying traffic based on network requirements.

Proxy Authentication Mechanism

- Proxy authentication refers to configuring the authentication method used by a proxy server and determining how client machines are validated when accessing proxies.
- By default, the proxy authentication feature is disabled, and it needs to be enabled to create new policies for users or groups.

How does it works

- The process of proxy authentication involves validating and verifying a client's request.
- Proxies act as access control devices, blocking requests until the user or client provides valid access credentials to the proxy.
- The HTTP proxy-authenticate header specifies the authentication method required to access resources through the proxy server.
- When a client communicates with a proxy server using the Hypertext Transfer Protocol (HTTP), additional information about the request and the client is sent in the HTTP request header.
- The proxy server responds with a header message in the HTTP response, requesting credentials for validation.

Two methods of authentication

- There are two methods of proxy authentication.

1.The first type is authentication using a username and password and

2. The other one is authentication through an IP address.

1.Authentication using a username and password:

- In this method, users are required to provide a username and

  password to authenticate their access to the proxy server.
- When signing up for proxy services, the provider typically assigns a username and password to the user.
- These credentials are then used to authenticate the user when making requests through the proxy server.
- The user includes the provided username and password in the proxy authentication headers or fields to establish their identity and gain access to the proxy resources.

2.Authentication through an IP address:

- With IP address authentication, the user's access to the proxy server is verified based on their IP address.
- This method involves configuring the proxy server to allow access only from specific IP addresses.
- Users need to provide their IP address to the proxy server administrator or proxy service provider.
- Once the IP address is registered and authorized, the user can access the proxy server without requiring a username or password. The proxy server identifies the user based on their IP address and grants access accordingly.