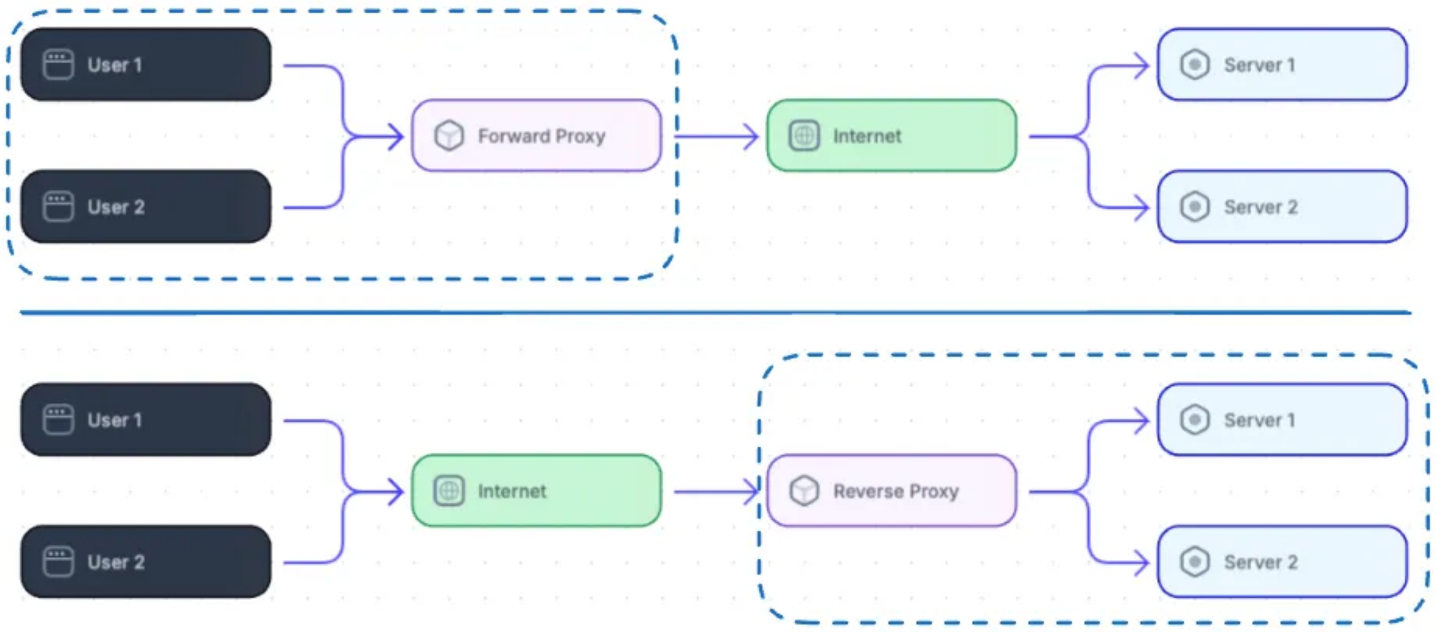
Proxy vs Reverse Proxy (Explained with Examples)

ASHISH PRATAP SINGH
OCT 30, 2024

241 9 15 Share

Proxies and **reverse proxies** are servers that sit between clients and servers to improve security, privacy and performance.

A **Proxy server** (sometimes called a **Forward proxy**) acts on behalf of clients, while a **Reverse Proxy** acts on **behalf** of servers.



Visualized using Multiplayer

In this article, we'll break down the key differences between **proxies** and **reverse proxies** and how they function with real-world examples and simple illustrations.

If you're enjoying this newsletter and want to get even more value, consider becoming a paid subscriber.

As a paid subscriber, you'll unlock all **premium articles** and gain full access to all premium courses on algomaster.io.

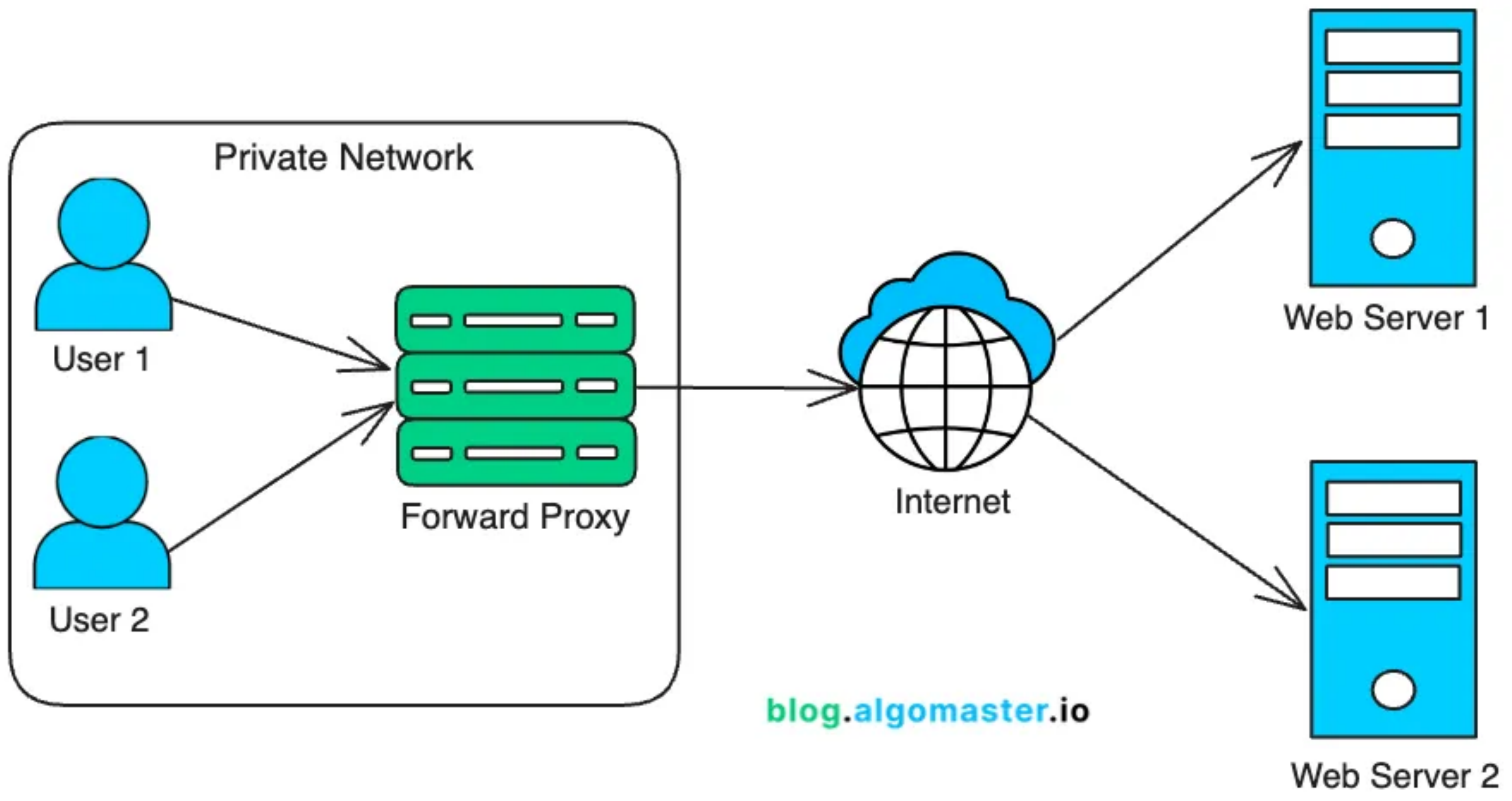
---

1. What is a Proxy Server?

> A **proxy** is an entity that has the authority to act on behalf of another.

In computer terms, a **proxy** (or a **forward proxy**) is a server that acts on behalf of clients on a network.

When you send a request, like opening a webpage, the proxy intercepts it, forwards it to the target server, and then relays the server’s response back to you.



Visualized using Multiplayer

Think of proxy server as a **middleman** that sits between a private network and the public internet.

Let's walk through a simplified example of how a proxy server handles a request:

1. The user types a website URL into their browser. The request is intercepted by the

proxy server instead of going directly to the website.

2. The proxy server examines the request to decide if it should forward it, deny it, or serve a cached copy.
3. If the proxy decides to forward the request, it contacts the target website. The website sees only the proxy server's IP, not the user's.
4. When the target website responds, the proxy receives the response and relays it to the user.

Key Benefits of Proxy Servers:

1. **Privacy and Anonymity:** Proxy servers hide your IP address by using their own, so the destination server cannot know your real location or identity.
2. **Access Control:** Organizations use proxies to enforce content restrictions, monitor internet usage.
3. **Security:** Proxies can filter out malicious content and block suspicious sites, providing an additional layer of security.
4. **Improved Performance:** Proxies cache frequently accessed content, reducing latency and improving load times for websites.

Is a VPN the same as a Proxy?

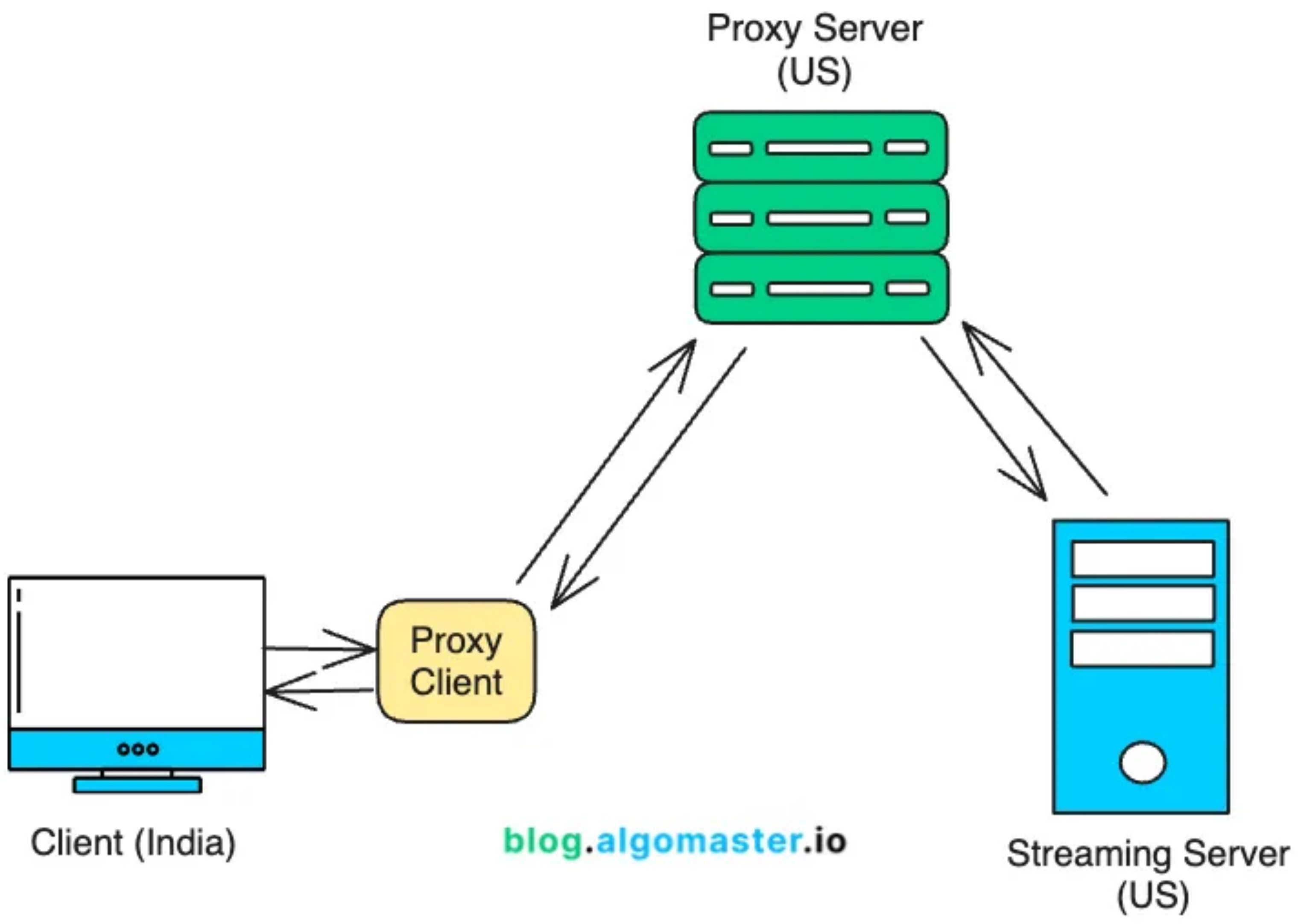
No. While both hide your IP, a **VPN** encrypts all your internet traffic, making it more secure. A proxy only forwards specific requests without necessarily encrypting them.

Real-World Applications of Proxy Servers

1. Bypassing Geographic Restrictions

One of the most common uses of proxy servers is **bypassing geographic restrictions** on websites and content.

Streaming services, for instance, often offer different content based on a user's location. With a proxy server based in the target region, you can access that region's content library as if you were a local user.



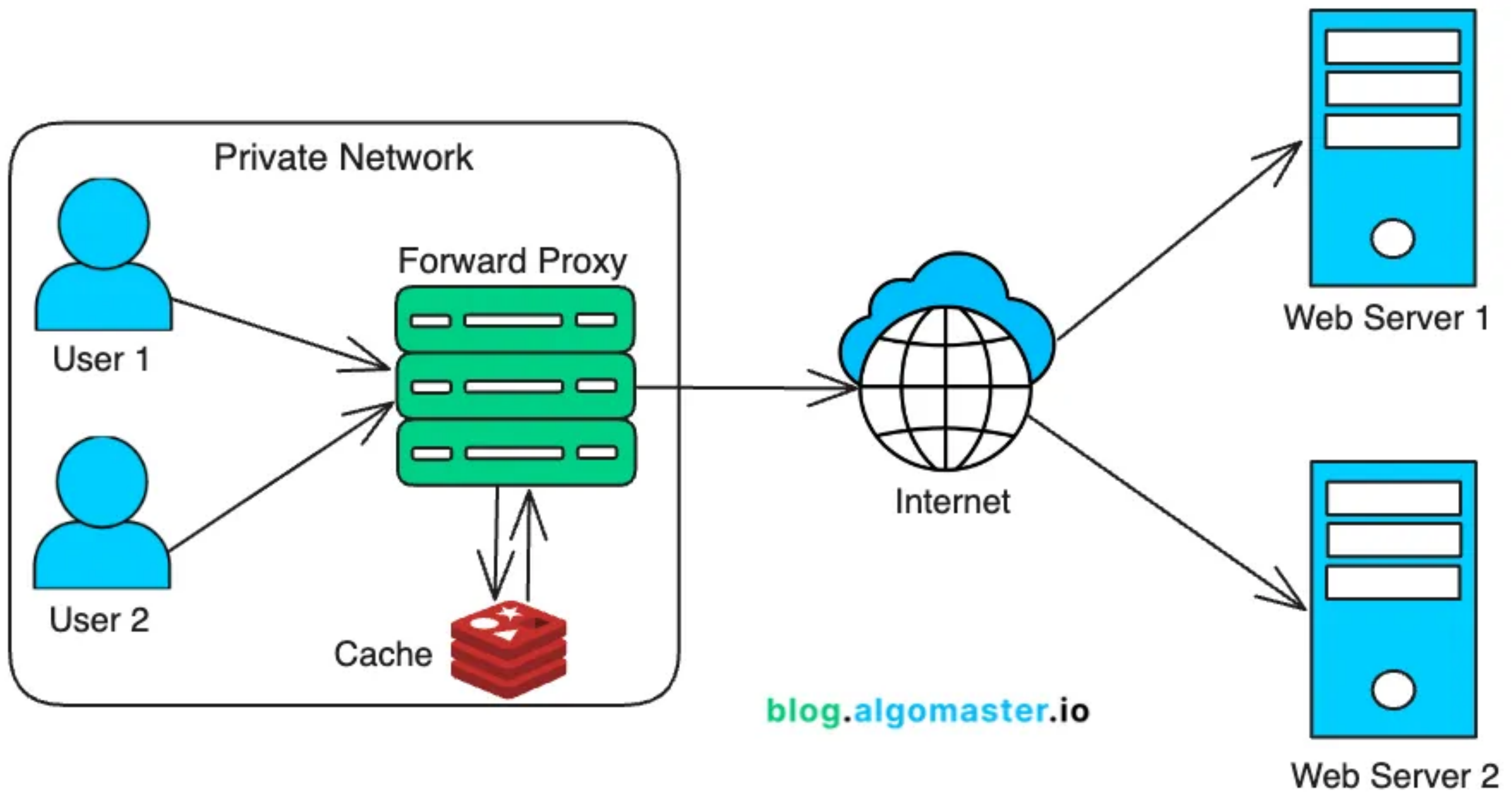
Visualized using [Multiplayer]

> **Example:** Suppose you’re in India and want to access the US library of a streaming

> platform (eg.. Netflix). By connecting to a proxy server located in the US, your request to the streaming platform will appear to be coming from the US, allowing access to its content as if you were a US-based viewer.

2. Speed and Performance Optimization (Caching)

Proxies can store cached versions of frequently accessed content, enabling faster load times and reducing bandwidth usage.



Visualized using Multiplayer

When a user requests cached content, the proxy server serves the stored copy rather than fetching it from the destination server, which reduces latency.

To avoid stale content, it uses a Time-To-Live (TTL) value, automatically expiring cached data after the configured time

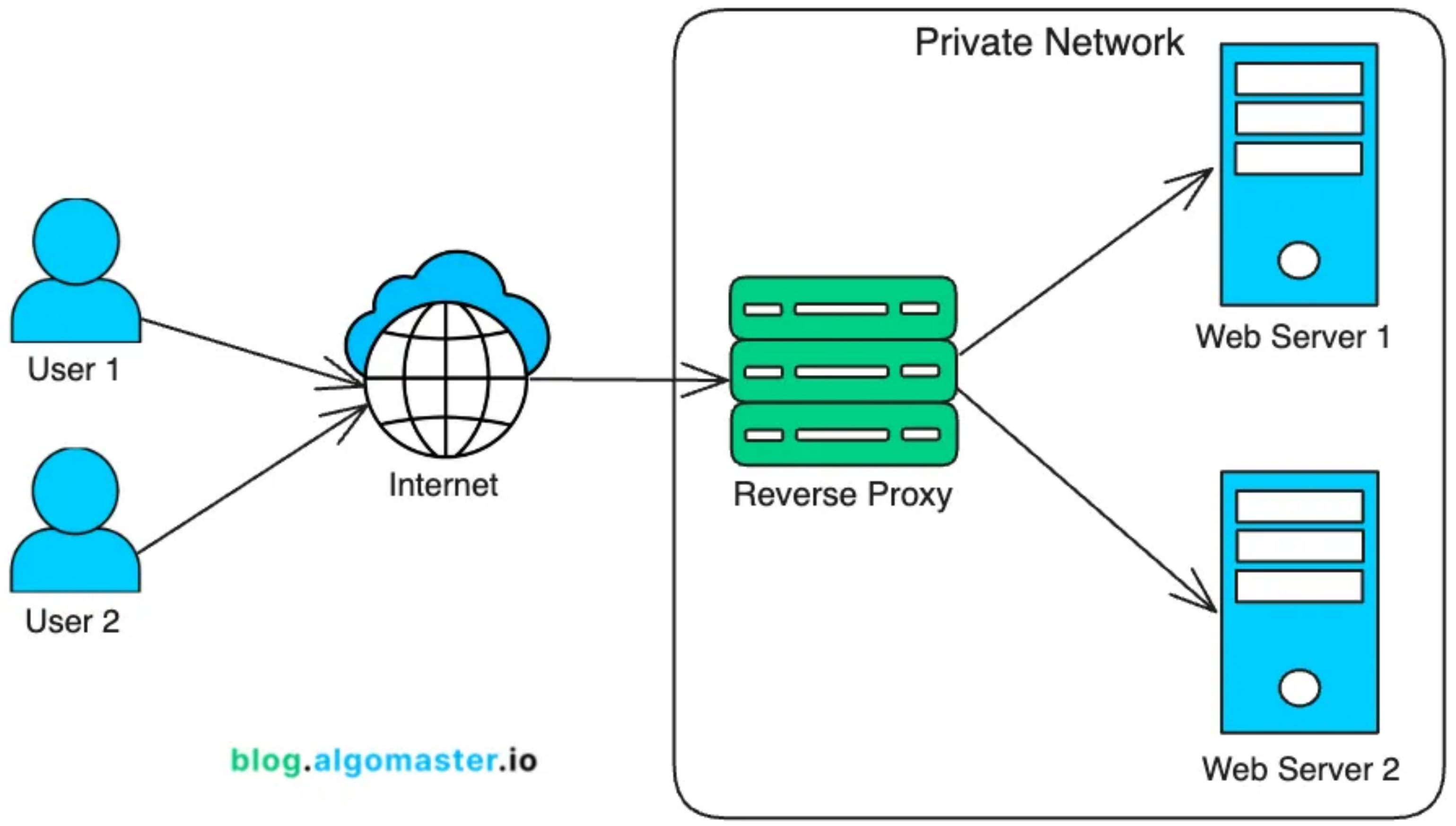
> **Example:** An organization with hundreds of employees frequently accessing the same online resources can deploy a caching proxy. This proxy caches common websites in it’s database, so subsequent requests are served quickly from the proxy’s storage, saving time and bandwidth.

---

2. What is a Reverse Proxy?

A **reverse proxy** is the **reverse** of a forward proxy. It regulates traffic coming into a network.

It sits in front of servers, intercepts client requests and forwards them to backend servers based on predefined rules.



Visualized using Multiplayer

Think of a reverse proxy as a **gatekeeper**. Instead of hiding clients from the server, it hides servers from clients.

Allowing direct access to servers can pose **security risks**, exposing them to threats like **hackers** and **DDoS attacks**.

A reverse proxy mitigates these risks by creating a single, controlled point of entry that filters and regulates incoming traffic all while keeping server IP addresses hidden.

With a reverse proxy in place, clients no longer interact directly with the servers. They only communicate with the reverse proxy.

Let’s walk through a simplified example of how a proxy server handles a request:

1. A user types a website URL into their browser, which sends a request to the server.
2. The reverse proxy server receives the request before it reaches the backend servers.
3. Based on predefined rules (like load balancing or server availability), the reverse proxy forwards the request to the appropriate backend server.
4. The backend server processes the request and sends a response back to the reverse proxy.
5. The reverse proxy relays the response to the client, with the client never directly interacting with the backend servers.

Key Benefits of Reverse Proxy

- **Enhanced Security:** By acting as a protective layer, a reverse proxy hides backend servers from clients, reducing the risk of attacks directly targeting backend infrastructure.
- **Load Balancing:** A reverse proxy can distribute incoming requests evenly across multiple backend servers, improving system reliability and preventing server overload.
- **Caching Static Content:** Reverse proxies can cache static assets like images, CSS, and JavaScript, reducing the need to fetch these files from the backend repeatedly.
- **SSL Termination:** Reverse proxies can handle SSL encryption, offloading this work from backend servers.
- **Web Application Firewall (WAF):** Reverse proxies can inspect incoming requests, acting as a firewall to detect and block malicious traffic.

Real-World Example of a Reverse Proxy

Cloudflare's reverse proxy is widely used by global websites and applications to boost speed, security, and reliability.

It's **Web Application Firewall (WAF)** and **DDoS protection** blocks malicious traffic before it reaches the site's servers, safeguarding against attacks and improving uptime.

Cloudflare's global content caching caches static and dynamic content at over 200 data centers around the world, storing frequently accessed files (like images, CSS, and JavaScript) closer to users. This significantly reduces load times and latency, as requests don't always need to travel to the origin server.

Setting Up a Reverse Proxy with Nginx

One of the most popular reverse proxy tools is **Nginx**.

Here's how you can set up a basic reverse proxy configuration using Nginx on a Linux server.

1. Install Nginx

```
sudo apt update sudo apt install nginx
```

2. Add Reverse Proxy Configuration

```
server {
    listen 80;
    location / {
        proxy_pass http://backend_server_ip;
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
    }
}
```

3. Test and Reload Nginx

```
sudo nginx -t sudo systemctl reload nginx
```

Load Balancing Across Multiple Servers

For a high-traffic website, spreading incoming requests across multiple backend servers is crucial.

A reverse proxy can implement load balancing algorithms such as round-robin, least connections, or IP hash, ensuring optimal distribution of traffic.

```
upstream backend_servers {
    ip_hash;
    server backend1.example.com;
    server backend2.example.com;
    server backend3.example.com;
}

server {
    listen 80;
    server_name example.com;

    location / {
        proxy_pass http://backend_servers;
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
    }
}
```

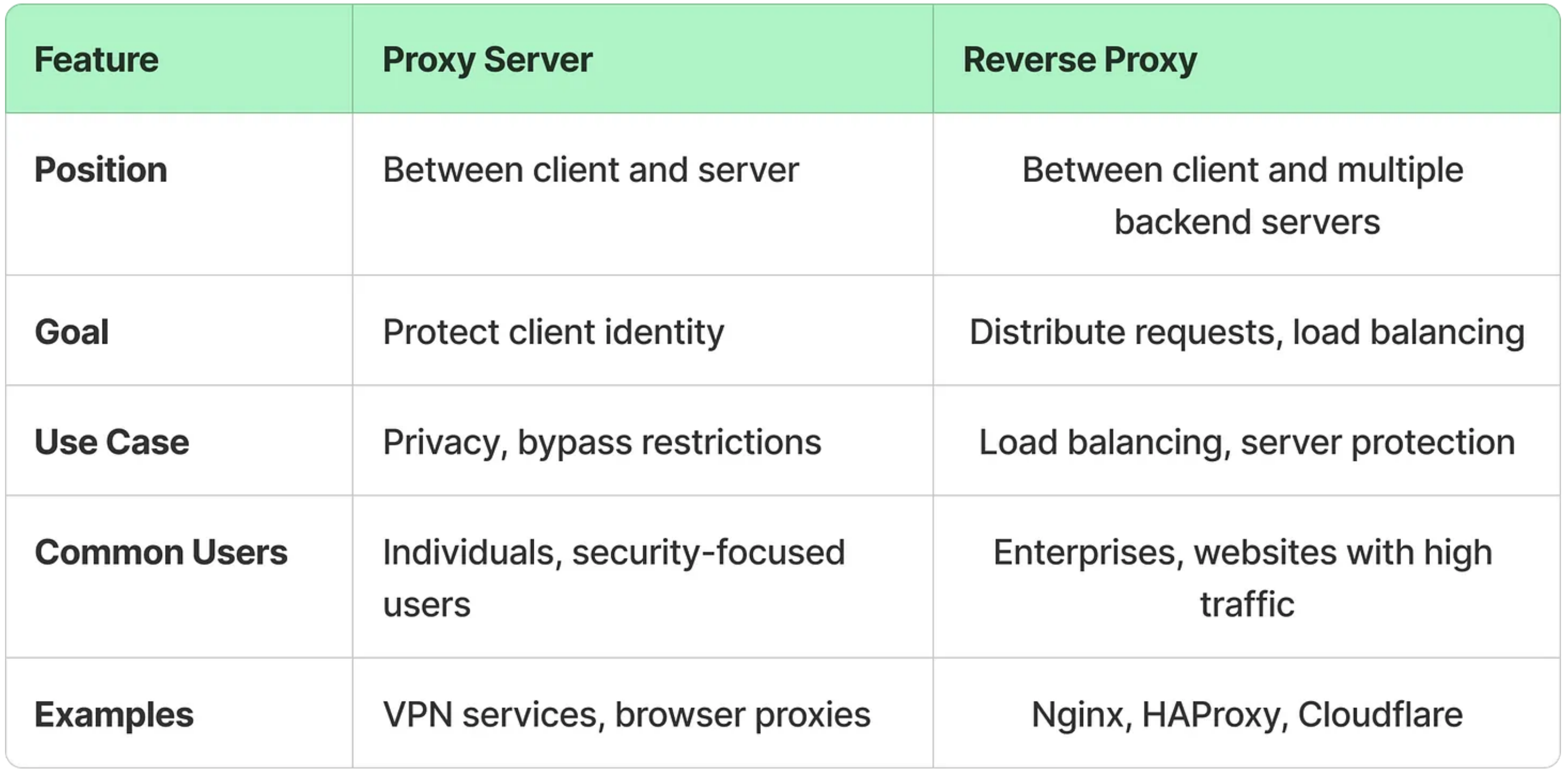
Nginx uses **round robin** by default. To change it, we can simply add the required algorithm (eg.. **ip_hash**) in the `upstream` block.

With this configuration, Nginx will balance requests among `backend1`, `backend2`, and `backend3`, ensuring no single server becomes overwhelmed.

---

3. Summary

Here’s a table summarizing the key details:



Feature	Proxy Server	Reverse Proxy
Position	Between client and server	Between client and multiple backend servers
Goal	Protect client identity	Distribute requests, load balancing
Use Case	Privacy, bypass restrictions	Load balancing, server protection
Common Users	Individuals, security-focused users	Enterprises, websites with high traffic
Examples	VPN services, browser proxies	Nginx, HAProxy, Cloudflare

---

Thank you for reading!

If you found it valuable, hit a like ❤️ and consider subscribing for more such content every week.

If you have any questions or suggestions, leave a comment.

This post is public so feel free to share it.

P.S. If you're enjoying this newsletter and want to get even more value, consider becoming a paid subscriber.

As a paid subscriber, you'll unlock all **premium articles** and gain full access to all premium courses on algomaster.io.

There are group discounts, gift options, and referral bonuses available.

Checkout my Youtube channel for more in-depth content.

Follow me on LinkedIn, X and Medium to stay updated.

Checkout my GitHub repositories for free interview preparation resources.

I hope you have a lovely day!

See you soon,
Ashish

241 Likes · 15 Restacks

← Previous

Next →

Discussion about this post

Comments Restacks

Write a comment...

Sketech | Unfiltered Dev Notes 4 Nov 2024

Liked by Ashish Pratap Singh

Great Post! Really good and clear explanations :-)

LIKE (4) REPLY SHARE

Venkatesh Mandapati 30 Oct 2024

Liked by Ashish Pratap Singh

Great demonstration 👍👍

LIKE (2) REPLY SHARE

1 reply by Ashish Pratap Singh

7 more comments...