Jahong Liu

System Design Basics 12: Proxies and Load Balancing

Proxies & Load Balancers

Proxies

Let's consider an analogy to understand the concept of a forward proxy server within the context of a network.

Let's imagine a situation where **you want to send a letter** to someone, but you **don't want the recipient to know your address**. So, you ask a **trusted friend** to send the letter on your behalf. Your friend takes your letter, puts it in a new envelope, and **writes their return address on it instead of yours**. Now, the recipient only sees your friend's address, not yours.

In the digital world, a **forward proxy server** acts like that trusted friend. When your computer (the sender) wants to access a website (the recipient), it sends the request to the proxy server (your friend). The proxy server then sends the request to the website on your behalf.

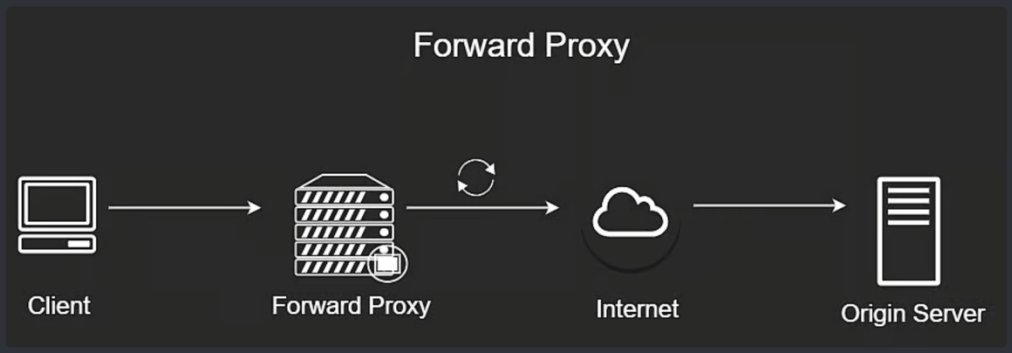
Like your friend, the proxy server masks your IP address and uses its own when communicating with the website, keeping your address hidden. If you frequently request the same data, the proxy server can cache (keep a copy of) the response and provide it to you without having to request the same data from the website again, which makes the process more efficient.

Moreover, a proxy server can filter your requests, much like how your friend might refuse to send certain types of letters that don't meet specific criteria. This is how a proxy server can restrict access to certain websites or resources, similar to how a company or school might block access to entertainment sites.

In summary, a forward proxy server serves as an intermediary in network communication

1. Providing privacy
2. Increasing efficiency through caching
3. Allowing control over network traffic

3. Allowing control over network traffic



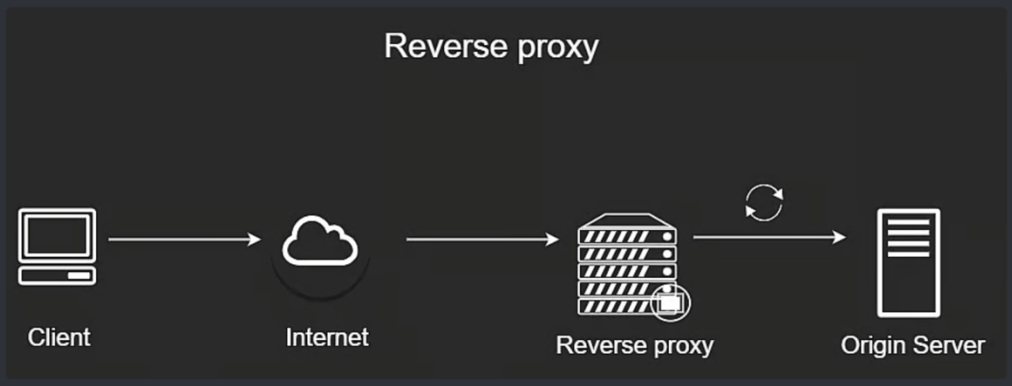
> In the diagram above the client is hidden from the origin server. The origin server only sees the proxy server's IP address.
> **Note**: Of course the internet is present at each step of the way, but this diagram is trying to convey that the client is *aware* that an extra network call is made by the proxy.

Reverse Proxy

A reverse proxy anonymizes the **destination server** instead of the client. It can also be considered as an inbound proxy.

Positioned between the internet and the server, a reverse proxy operates differently from a forward proxy. Instead of receiving requests from clients and forwarding them, it receives requests and forwards them to the appropriate server. The key benefit of a reverse proxy is its ability to protect servers by managing incoming requests and distributing the load across multiple servers. Furthermore, it provides protection against Distributed Denial of Service (DDoS) attacks, safeguarding websites.

While a forward proxy acts as an intermediary between the client and the internet, offering an additional layer of security and privacy for clients, a reverse proxy serves a similar purpose for the server. A prime example of a reverse proxy is a Content Delivery Network (CDN). Instead of clients directly accessing the origin server, the reverse proxy (CDN) handles the requests. If the requested content is available, the reverse proxy serves it directly to the client, effectively reducing latency and alleviating the load on the origin server.



> In the diagram above the server is hidden from the client. The client does not know which server *actually* fulfilled the request.
> **Note**: Of course the internet is present at each step of the way, but this diagram is trying to convey that the client is only *aware* of the network call between the client and proxy, and *not* the call between the reverse proxy and server.