Route53 and Failover

Amazon Route 53 is a scalable Domain Name System (DNS) web service that translates domain names into IP addresses and routes users to endpoints, turning a user-friendly URL like `www.example.com` into a numeric IP address like `192.0.2.1`, usually representing an EC2 instance, a load balancer, or another type of endpoint or AWS resource.

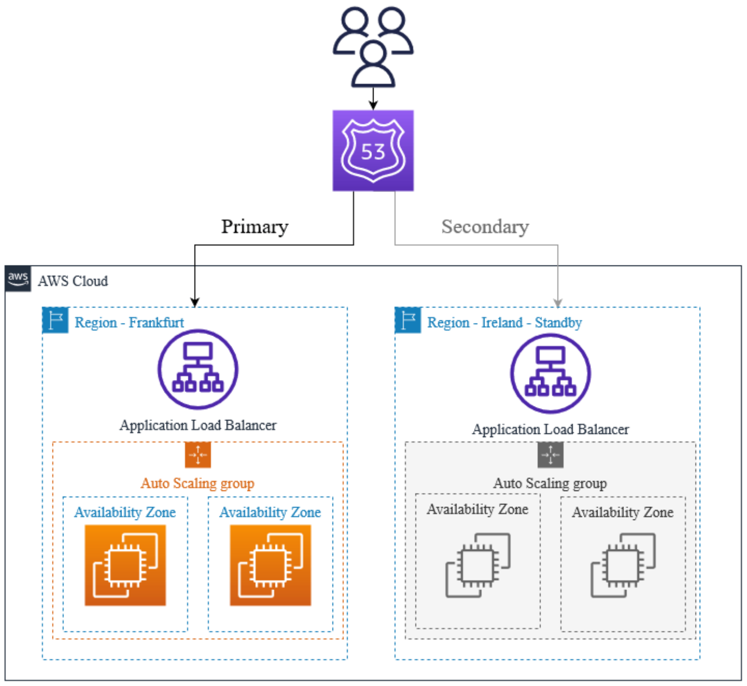
The service also offers a myriad of routing policies, capable of directing traffic to the necessary IP address based on specific criteria and conditions. A brief summary of the different routing policies is given below:

- **Simple Routing**: This is the most straightforward option, where Route 53 returns a single resource record set for a domain name. It's useful when you want to direct traffic to a single endpoint.
- **Failover Routing**: Failover routing allows us to configure primary and secondary endpoints and switch between them. If the primary endpoint becomes unhealthy (as determined by health checks), Route 53 automatically routes traffic to the secondary endpoint, ensuring minimal downtime.
- **Latency-Based Routing**: This policy routes traffic to the endpoint that provides the lowest latency for the user, enhancing performance by directing users to the nearest or fastest server.
- **Geolocation Routing**: This option lets you route traffic based on the geographic location of the user. You can specify different endpoints for users in different regions, which can be helpful for compliance or to provide localized content.
- **Geoproximity Routing**: Similar to geolocation routing, this policy allows you to route traffic based on the user's location and the location of your resources. You can also specify bias to favor certain endpoints.

- **Weighted Routing**: This policy allows you to distribute traffic across multiple endpoints based on assigned weights. For example, you might route 70% of traffic to one endpoint and 30% to another, useful for gradual rollouts or A/B testing.
- **Multi-Value Answer Routing**: This option allows Route 53 to return multiple IP addresses in response to a single DNS query. Clients can then choose which address to connect to, which can enhance redundancy and load balancing.

Now, while a more in-depth summary of all the different routing policies is beyond the scope of this book, we will be taking a closer look at Failover Routing in this section, with Geoproximity and Geolocation routing being the subjects of the next section.

Failover routing, simply put, allows us to direct traffic to one particular path when a resource (designated as the primary resource) is healthy and to direct it through another path when it is not. The resource that Amazon Route53 redirects to when the primary resource is unavailable is also called the secondary resource.



Failover Routing with Primary and Secondary Resource **(Source: StormIT)**

Failover routing furthermore, has two different configuration modes, each with its own modus operandi: **Active-Active Failover** and **Active-Passive Failover**

Active-Active failover in Route 53 allows traffic to be distributed across multiple endpoints, all of which are capable of handling requests simultaneously. This setup improves availability and responsiveness by distributing traffic among healthy endpoints.

Active-Passive failover, on the other hand, uses a standby endpoint that only becomes active when the primary endpoint fails. This configuration ensures minimal downtime by automatically redirecting traffic to the standby endpoint when the primary endpoint becomes unavailable.

Therefore, we use the Active-Active failover configuration when we want all of your resources to be available the majority of the time while the Active-Passive failover configuration is to be used when we wish that the secondary resource serve us mostly in a reserve capacity, being utilized only in cases where they system is facing significant disturbances.

TLDR;

Route 53 is a DNS web service with support for failover policies that enable robust fault tolerance and high availability for applications, helping ensure a seamless user experience even during moments of endpoint failure.