1. Distributed web infrastructure

Three-Server Web Infrastructure for foobar.com

Building upon the single-server model, let's enhance scalability and reliability with a three-server architecture:

Components:

Server 1: Application Server (running your codebase)

Server 2: Primary Database Node (MySQL Master)

Server 3: Replica Database Node (MySQL Slave)

Web Server (Nginx): Remains on Server 1 for simplicity

Load Balancer (HAproxy): Dedicated server (optional)

1. Additional Elements:

Load Balancer: Distributes incoming traffic across the application servers (Server 1) for improved performance and scalability. We'll use HAproxy with a round-robin distribution algorithm, where requests are sent to servers in a cyclical order, ensuring even distribution.

Database Cluster (Primary-Replica): Enhances database availability and performance. The Primary node (Server 2) handles all writes and acts as the main source of truth. The Replica node (Server 3) receives synchronized copies of data from the Primary, offering faster read access and serving as a failover option.

2. Explanations:

Adding Load Balancer: Improves scalability by handling increased traffic and avoids Single Point of Failure (SPOF) by not relying on a single application server. Round-robin distribution ensures fair sharing of requests.

Active-Passive vs. Active-Active: In this setup, we employ an Active-Passive configuration, where only the Primary node (Server 2) receives writes. The Replica (Server 3) passively synchronizes data and takes over only if the Primary fails. For continuous read/write availability, consider Active-Active, where both nodes handle writes but require more complex configurations.

Primary-Replica Cluster: The Primary node (Server 2) is the authoritative source of data and handles all write requests. The application server primarily interacts with the Primary for writes operations. The Replica node (Server 3) offers faster read performance and serves as a backup if the Primary fails. The application might read from either node, leveraging the read-scaling potential.

Difference between Primary and Replica Nodes: The primary node accepts write operations and is responsible for ensuring data consistency and integrity. The replica nodes replicate data from the primary and serve read requests, offloading read traffic from the primary node. The primary node is critical for the application's write operations, while replica nodes enhance read performance and provide redundancy.

3. Potential Issues:

SPOF: While the infrastructure is more resilient than the single-server model, potential SPOF include the load balancer and individual server hardware. Consider redundancy measures for these components.

Security: Implement firewalls on all servers, enable HTTPS for encrypted communication, and follow security best practices to protect against breaches.

Monitoring: Integrate monitoring tools to track server health, performance metrics, and resource utilization. Monitor application logs for errors and security events.

4. Improvements:

- Implement firewall protection on all servers.
- Configure HTTPS for secure communication between the browser and server.
- Integrate monitoring tools for system health, performance, and security.
- Consider deploying the load balancer on a separate server for further redundancy.
- Explore clustering the application servers for even greater scalability and high availability.