AWS Reference Architectures

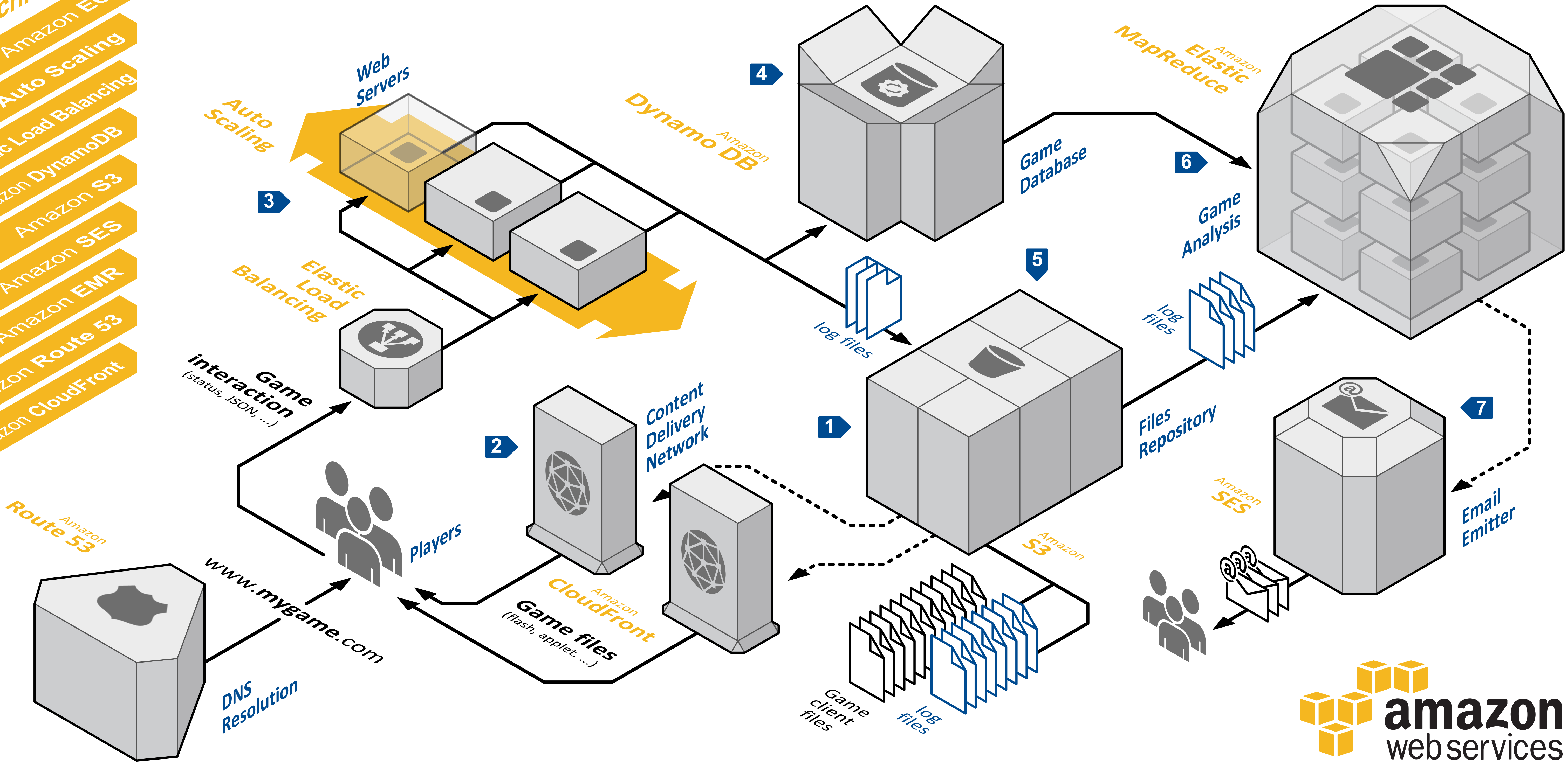
- Amazon EC2
- Auto Scaling
- Elastic Load Balancing
- Amazon DynamoDB
- Amazon S3
- Amazon SES
- Amazon EMR
- Amazon Route 53
- Amazon CloudFront

ONLINE GAMES

Online games back-end infrastructures can be challenging to maintain and operate. Peak usage periods, multiple players, and high volumes of write operations are some of the most common problems that operations teams face.

But the most difficult challenge is ensuring flexibility in the scale of that system. A popular game might suddenly receive millions of users in a matter of hours, yet it must continue to provide a satisfactory player experience. Amazon Web Services provides different tools and services that can be used for building online games that scale under high usage traffic patterns.

This document presents a cost-effective online game architecture featuring automatic capacity adjustment, a highly available and high-speed database, and a data processing cluster for player behavior analysis.



amazon web services

System Overview

1 Browser games can be represented as client-server applications. The client generally consists of static files, such as images, sounds, flash applications, or Java applets. Those files are hosted on **Amazon Simple Storage Service** (Amazon S3), a highly available and reliable data store.

2 As the user base grows and becomes more geographically distributed, a high-performance cache like **Amazon CloudFront** can provide substantial improvements in latency, fault tolerance, and cost. By using **Amazon S3** as the origin server for the **Amazon CloudFront** distribution, the game infrastructure benefits from fast network data transfer rates and a simple publishing/caching workflow.

3 Requests from the game application are distributed by **Elastic Load Balancing** to a group of web servers running on **Amazon Elastic Compute Cloud** (Amazon EC2) instances. **Auto Scaling** automatically adjusts the size of this group, depending on rules like network load, CPU usage, and so on.

4 Player data is persisted on **Amazon DynamoDB**, a fully managed NoSQL database service. As the player population grows, **Amazon DynamoDB** provides predictable performance with seamless scalability.

5 Log files generated by each web server are pushed back into **Amazon S3** for long-term storage.

6 Managing and analyzing high data volumes produced by online games platforms can be challenging. **Amazon Elastic MapReduce** (Amazon EMR) is a service that processes vast amounts of data easily. Input data can be retrieved from web server logs stored on **Amazon S3** or from player data stored in **Amazon DynamoDB** tables to run analytics on player behavior, usage patterns, etc. Those results can be stored again on **Amazon S3**, or inserted in a relational database for further analysis with classic business intelligence tools.

7 Based on the needs of the game, **Amazon Simple Email Service** (Amazon SES) can be used to send email to players in a cost-effective and scalable way.