LearNet

C2: Mitul Saha, Nakul Goenka, Jason Tang
18-500 Capstone Design, Spring 2022
Electrical and Computer Engineering Department
Carnegie Mellon University

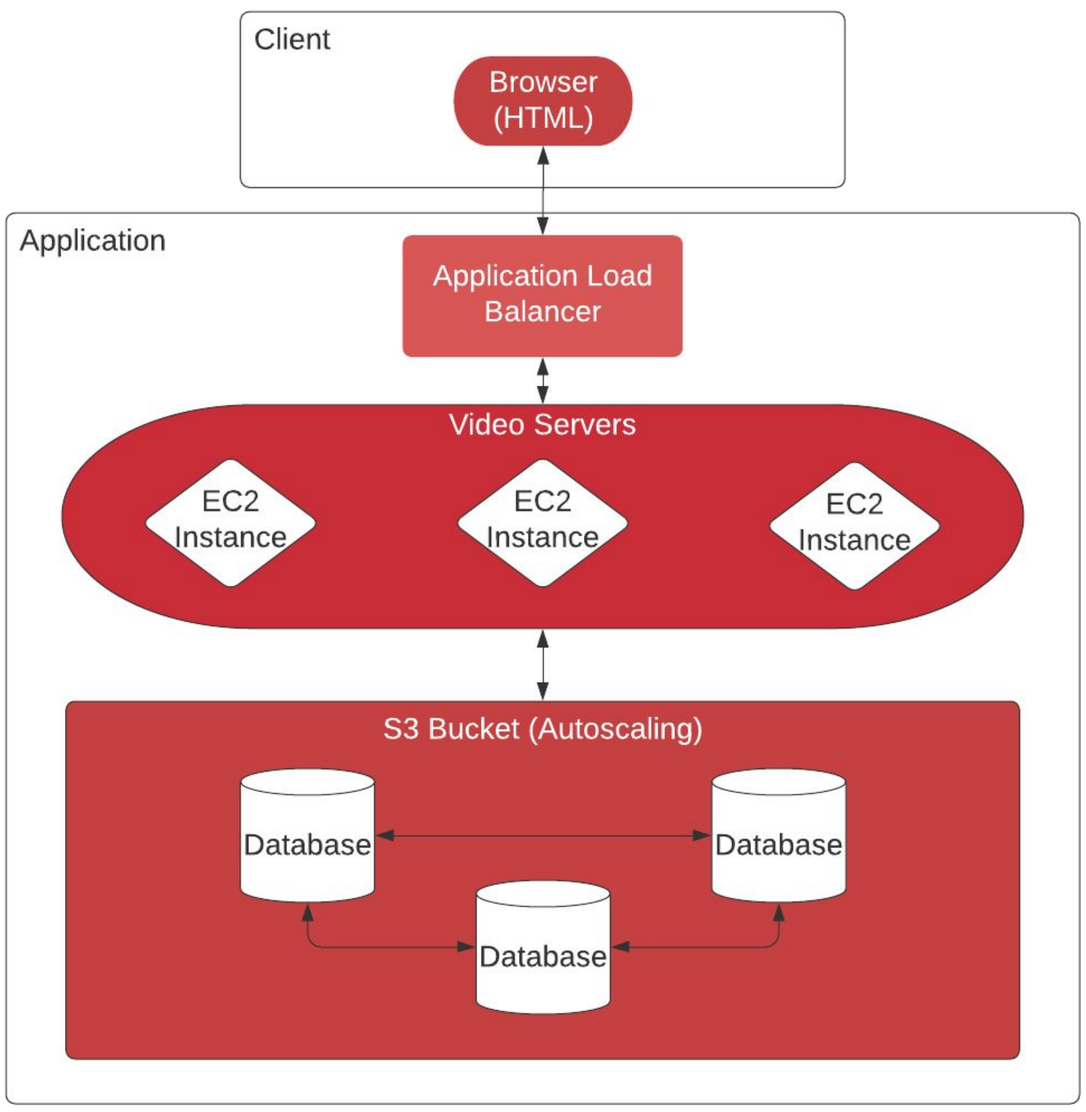
Electrical & Computer ENGINEERING
Carnegie Mellon

Product Pitch

Typical server architecture is increasingly moving towards multi-tier systems that rely on load balancing to meet storage and workflow efficiency needs. Though 3rd-party load balancing solutions are available, they function as black-boxes may not meet specific system demands. Furthermore, there is a lack of public knowledge on custom load balancing for server-specific solutions. We document performance of some traditional and some custom dynamic load balancing algorithms within a real-world system: a video streaming server.

System Architecture

Clients will have videos streamed in many 1MB chunks requests to the video app. Each request goes to a proxy server that load balances by choosing a video server with a specific algorithm. The video server retrieves the requested chunk from the AWS S3 database and sends it to the client while sending relevant server metrics to the load balancer.



Conclusions & Additional Information

Scan QR or follow this link for more information: https://course.ece.cmu.edu/~ece500/projects/s22-teamc2/

We made two key design tradeoffs. Firstly, our load balancing proxy now uses server metrics to make decisions instead of user metrics. These values are less useful but more reliable and better adhere to the server-client contract. We also streamlined our server architecture. This corresponds less to commercial video streams but makes load balancing decisions more impactful.

System Description

Algorithm	Parameters	Brief Description
Round Robin	N/A	Chooses server in sequence (e.g. 1, 2, 3, 1, 2, 3, …)
e-greedy	response time, network I/O, e value	Chooses best performing server with chance ‘e’ and otherwise random
Soft-UCB adaptation	response time, network I/O, p value	Chooses best performing server and worsens its metric by fixed value ‘p’ each time chosen
Hard-UCB adaptation	response time, network I/O, k value	Chooses best performing server from list of k-least recently chosen servers

4 Different Use Cases (VM Groups)	
1	Same Server Location, Same Server Specifications
2	Same Server Location, Different Server Specifications
3	Different Server Locations, Same Server Specifications
4	Different Server Locations, Different Server Specifications

System Evaluation

Metric	Benchmark	Description
User Metrics		
User Latency	2s	Time elapsed between the video request from the user and receiving video data from the server
Bit Rate	Video-based	How many bits are transmitted over a specified time
Load Balancer Metrics		
LB Latency	500 ms	Time elapsed between the LB receiving the request until a response from the video server is received
CPU Utilization		Processor utilization (%) of a server/VM (async)
Network I/O		Data volume (bytes) of input/output to a server (async)

Our custom load balancing algorithms will use the load balancer metrics for their decision-making process. JMeter user scripts are generated and ran through BlazeMeter to obtain the user metrics between algorithms which are then graphed and compared to determine their efficacy in different use cases.