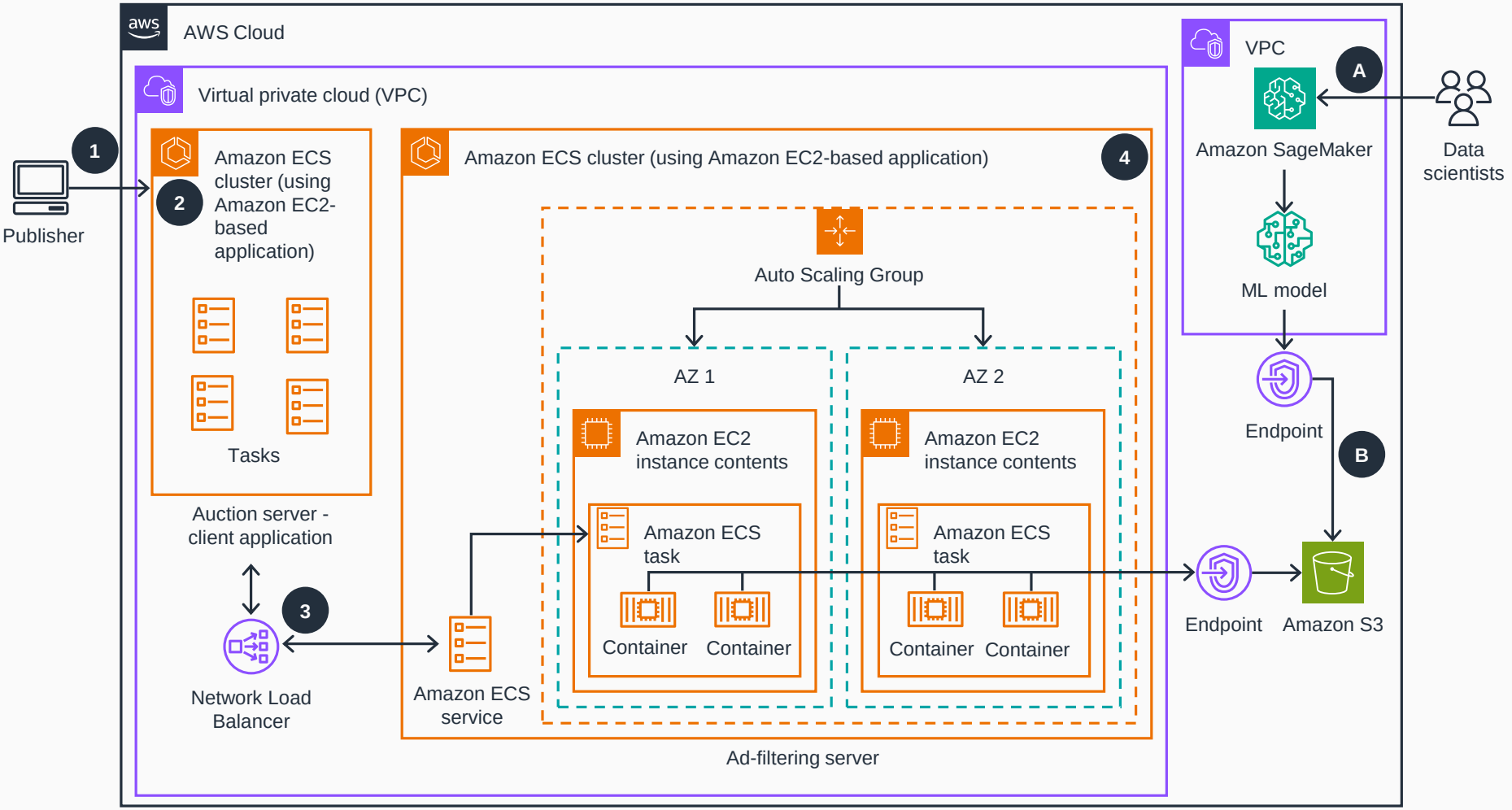
Guidance for Low-Latency High-Throughput Model Inference Using Amazon ECS

This architecture diagram shows how to build a real-time ad-filtering solution that can serve millions of requests per second by hosting the solution's ML model on Amazon ECS.



Data Scientist

A Data scientists use **Amazon SageMaker** to experiment with, build, and train their ML model. Once the model is ready, it is saved in **Amazon Simple Storage Service (Amazon S3)**.

B The trained model is read and loaded by the **Amazon Elastic Container Service (Amazon ECS)** model inference task. The model is hosted as a Thrift endpoint. Incoming requests, in OpenRTB format (for real-time bidding), are used for inference.

Publisher

1 A publisher issues requests to a supply-side platform (SSP) auction server for an ad placement.

2 The auction server (a client application) is hosted as an **Amazon ECS** application within the SSP's virtual private cloud (VPC). The auction request issues a bid request based on the OpenRTB format.

3 **Network Load Balancer** distributes the incoming requests to an **Amazon Elastic Compute Cloud (Amazon EC2)**-based **Amazon ECS** cluster that hosts the ad-filtering ML server. The purpose of the ad-filtering ML server is to infer the likelihood of a bid for every bid request, filtering the demand partners that need to be sent to the auction request, and optimizing the cost per bid.

4 The ad-filtering ML server is hosted as a container within an **Amazon EC2-based Amazon ECS cluster**. An **Amazon EC2 Auto Scaling** group maintains the desired number of **Amazon EC2** instances running across multiple Availability Zones (AZs) to maintain high availability. **Amazon ECS** deploys and maintains the desired capacity of the **Amazon ECS** tasks, hosting the ML container. Each task loads the ad-filtering model from an **Amazon S3** bucket and hosts it as a Thrift protocol–based endpoint. This helps in low-latency-based communication, and multiple instances of the tasks support a high number of concurrent requests.

Reviewed for technical accuracy January 24, 2024

AWS Reference Architecture