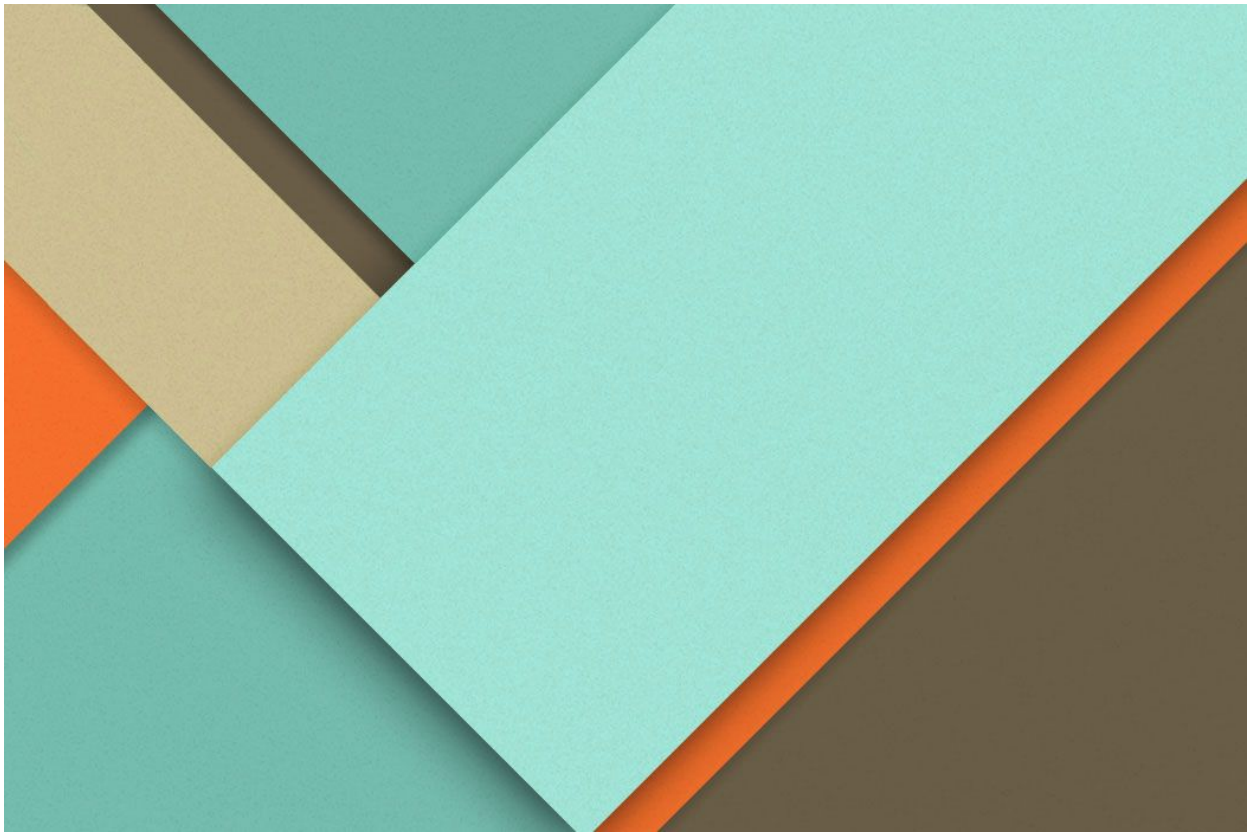


Bloud

07.31.2019

Joebert S. Jacaba
CEO, Kenchlightyear
1602 Valderrama Tower, Sumulong, Sto. Nino
Marikina City, Philippines

Overview

Bloud is a blogging system native to the cloud. The company prefers to use AWS being the de facto cloud leader. This is my response to the directive of the CTO to prepare a design to achieve the project goals.

Goals

1. Highly scalable
2. Cost effective
3. Highly available
4. High Performance
5. Secure
6. Users are coming from different continents

Specifications

- Create new posts on the system with text, images, audio and videos
- Uploaded images are converting into multiple sizes while video and audio are transcoded into multiple versions to support different devices and internet bandwidths
- Login using social accounts such as Facebook, Google, Twitter
- Users must be able to leave comments and like posts
- The system should support low end devices and limited internet speeds

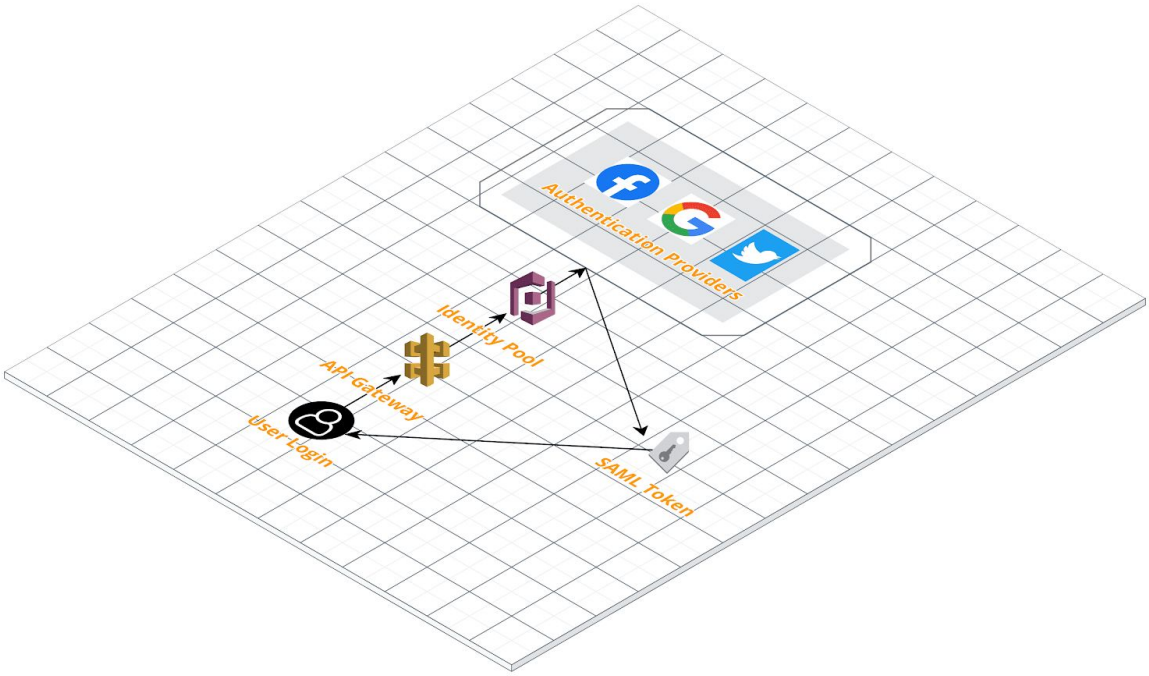
Solution

Use AWS and leverage Serverless technology to maximize scalability, cost savings, availability, and performance. API Gateway, Lambda, S3, AWS Elemental MediaConvert and DynamoDB will be the major components. Security will be applied on all levels. IAM, ACM, WAF and KMS will be used in this area. Application will be available globally and will perform uniformly if accessed from any location. Route53, Cloudfront will handle this portion.

Architecture

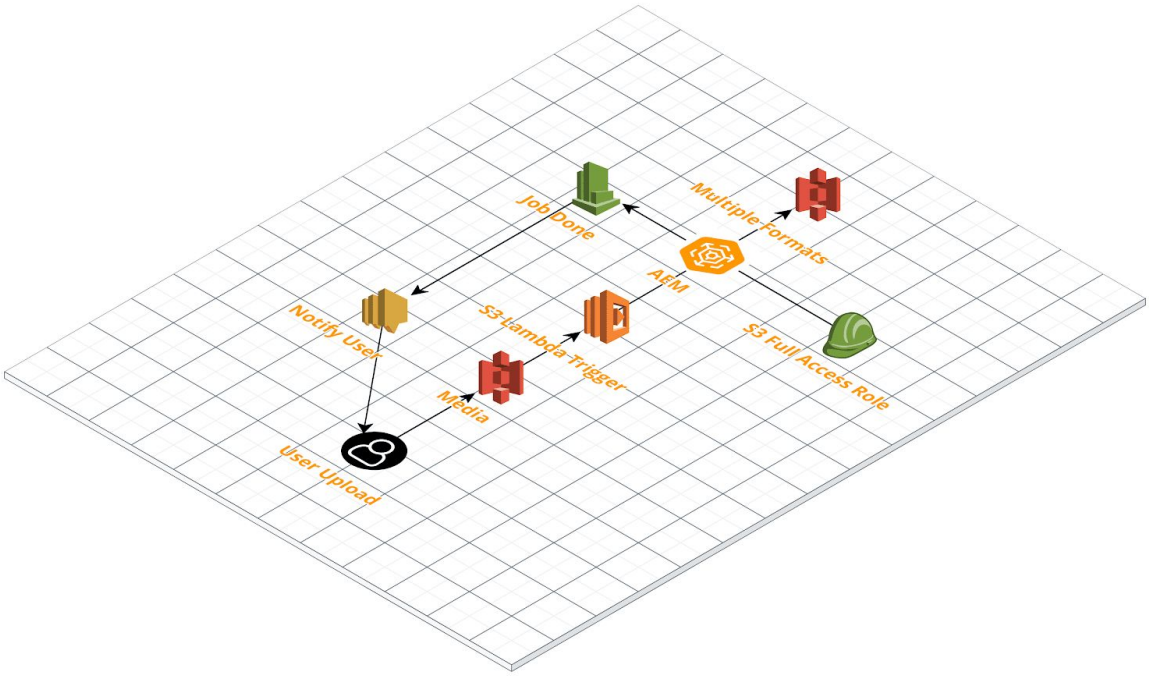
I. Authenticating Users

Cognito will be used to authenticate users. Federated Identities for Facebook, Google and Twitter will allow any user with existing accounts to access the system. The SAML Token returned by the Authentication Providers will be used in the access-protected areas of the API Gateway to allow or deny access.



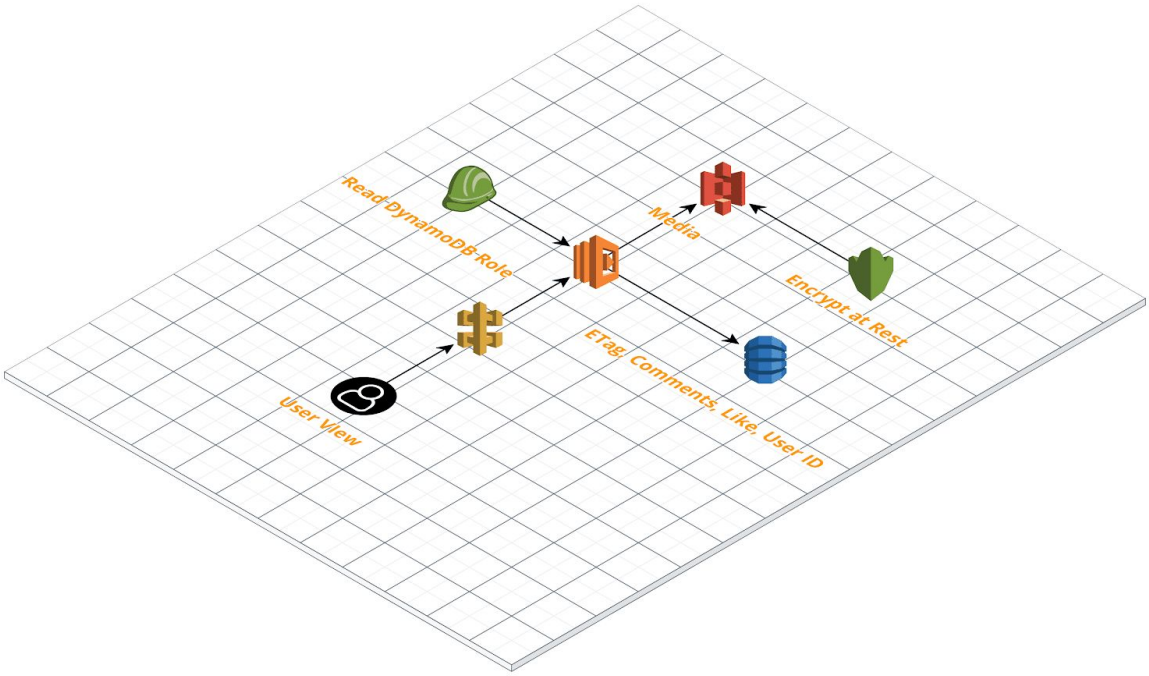
II. New Blog Entry

The text portion of the blog will be stored on S3 as a document object. All other media files will be stored on S3 as well. S3 will have a trigger on PutItem and will invoke a Lambda function. AWS Elemental MediaConvert will produce multiple formats for all media uploaded. One of the formats must be friendly to low end devices and those with limited internet speed.



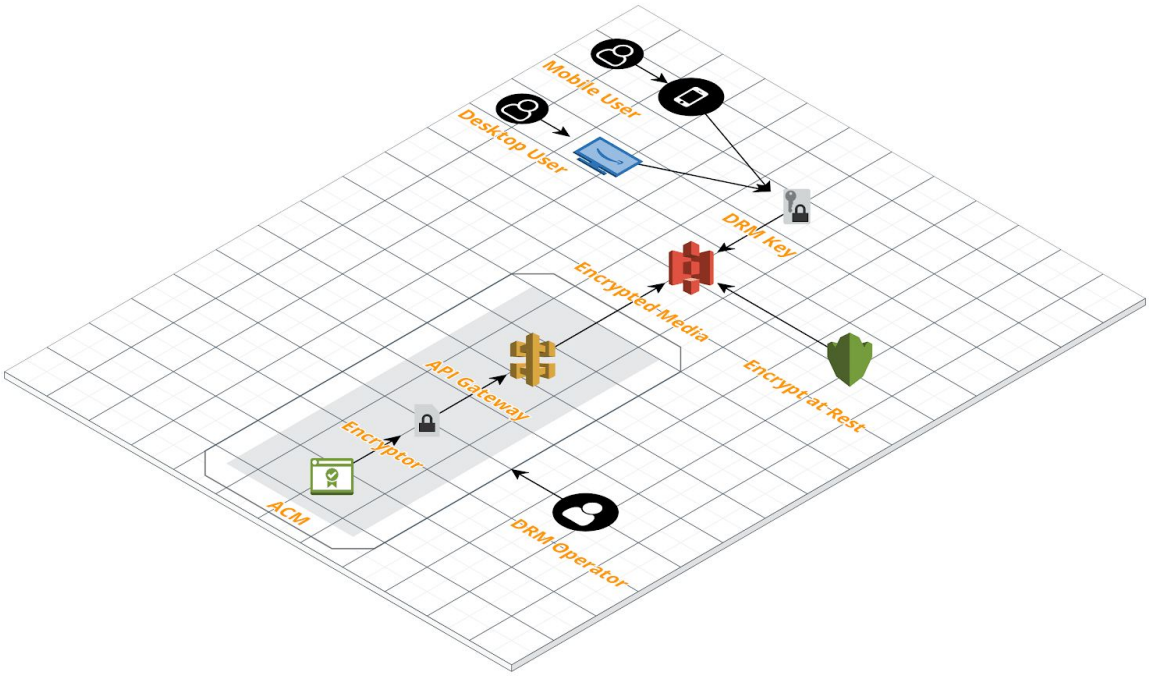
III. Viewing a Blog Entry

When a user views a blog entry, a Lambda with read-only access to the document and media files on S3 will serve the request. Comments and likes will be stored in DynamoDB associated to ETag of S3 and the User ID. Here we implement IAM Role to limit the access of Lambda.



IV. Server Side and Distribution Encryption

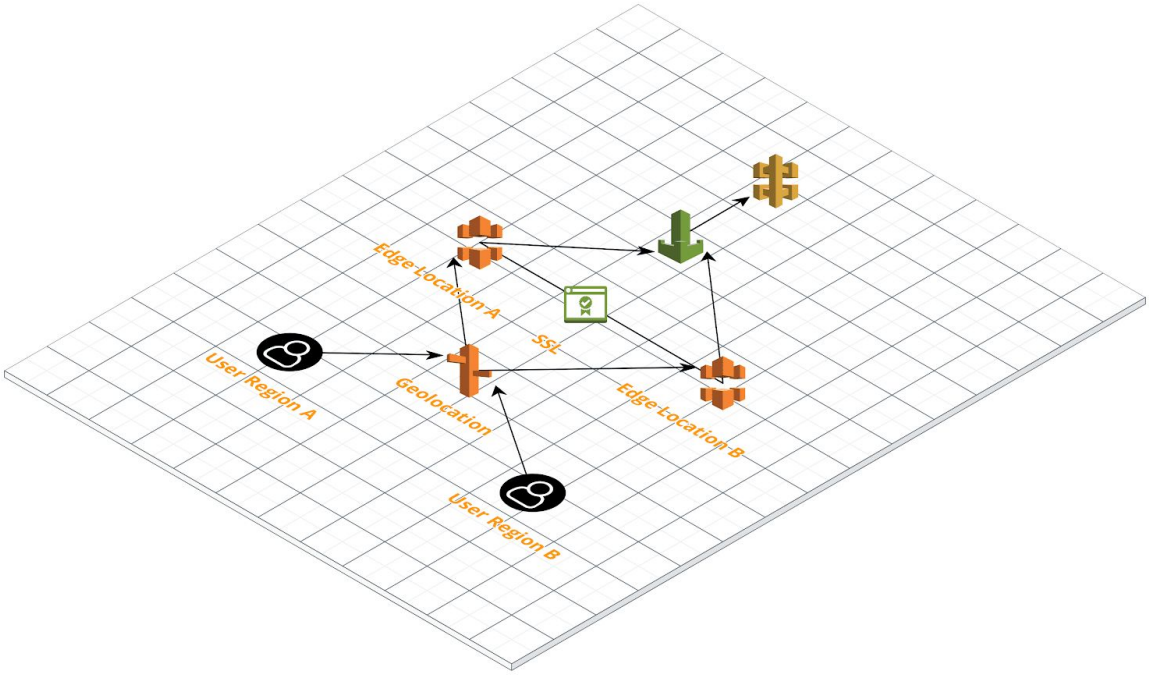
We use ACM to generate certificates to act as encryptors. Our DRM operators will encrypt media files on S3 to ensure that only users of the system can view, listen or play media files. In addition KMS will be used to encrypt all S3 data at rest. Multiple viewing devices should be supported as well.



V. Global Reach

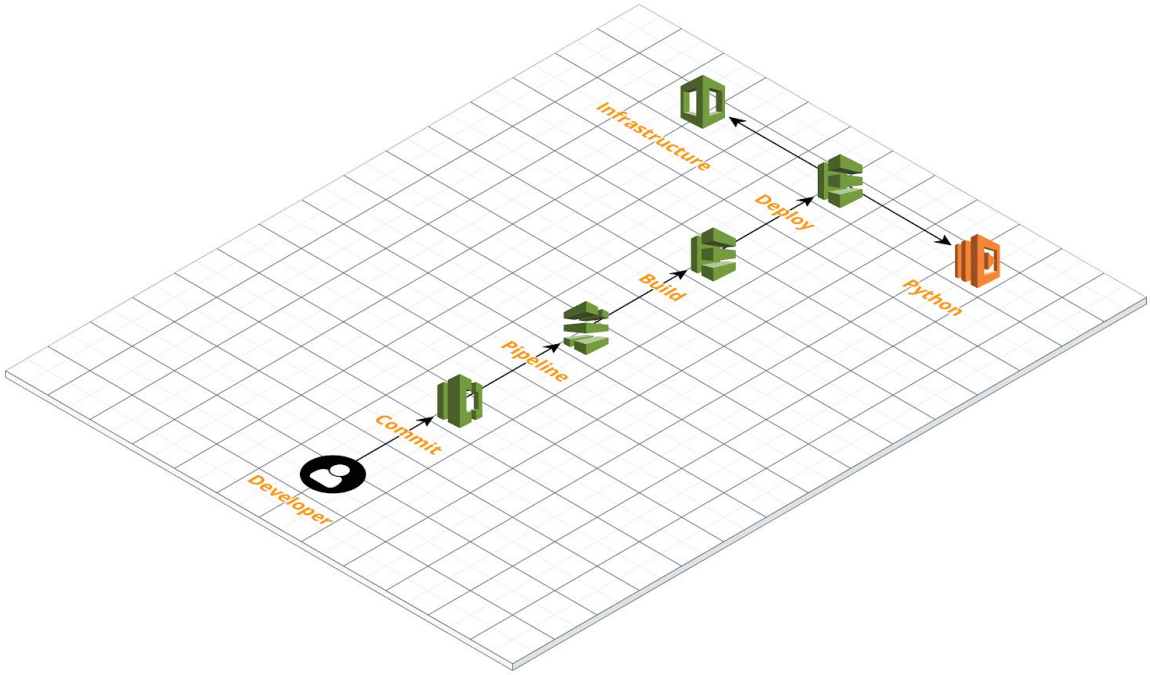
Route53 with Geolocation Routing Policy will direct users to Edge Locations appropriate to them. We will also protect our API Gateway with SSL from ACM and with WAF. All other resources that does not need to be publicly accessible will

remain in private zones.



CICD

Both Developers and Operations will follow a CICD strategy. Here CodeCommit, CodePipeline and CodeDeploy wii be used. Infrastructure code will be on Cloudformation. Python will be used as code for Lambda functions.



Feedback

Please revert to me any questions, comments and suggestions so I can further improve this document.