Access Management in Cloud Networks

Comparing OAuth 2.0-Based Services on Securing Serverless Applications

Hannes Larsson

Introduction

With the introduction of cloud technology multiple applications have been moved to a cloud architecture, allowing companies to more easily deploy, manage and expand software. However, as cloud technology is a relatively modern solution it is not extensively researched yet, leaving applications vulnerable to wider attack surfaces and unknown risks. As such, companies with services deployed on the cloud are required to maintain a more strict level of security and should always strive to follow the latest best practices in cloud development.

This thesis presents a comparison of Cognito and Curity, two services that have implemented the OAuth 2.0 protocol, in order to identify when Curity might be advantageous to the use of cloud-native Cognito when securing access in serverless applications. This comparison is done in the cloud infrastructure AWS.

Problem & Methodology

In order to compare Cognito and Curity, the following problems had to be answered:

1. What are key differences between Cognito and Curity?
2. How should Cognito and Curity be evaluated when deciding which is better att securing serverless applications?
3. In what situations would Curity be advantageous to Cognito?

The thesis approaches these problems by gathering and summarizing information about Cognito and Curity, analyzing the gathered data and developing use cases in order to compare the capabilities of Cognito and Curity in an iterative process. Areas for comparison are security features, OAuth 2.0 capabilities, OIDC capabilities, price and flexibility.

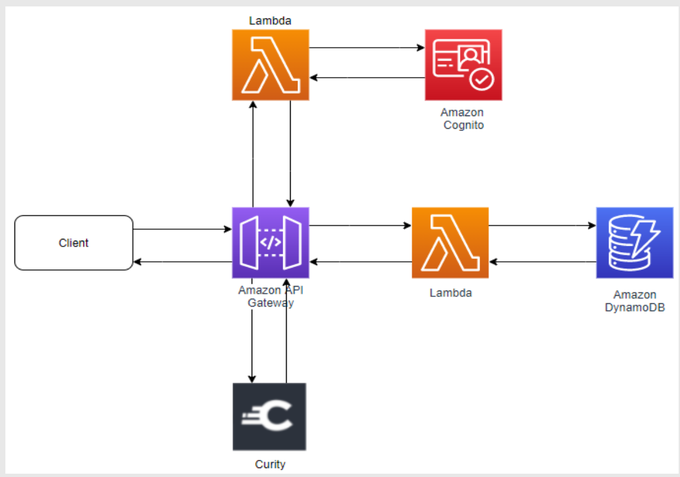
Results

The results gathered in the thesis presents differences between Curity and Cognito, with Curity having a wider support for OAuth and OIDC capabilities such as introspection flow, DCR and hybrid flow, a larger array of security features such as geolocation tables and a logarithmic scaling in price whereas Cognito has a more linear scaling in price.

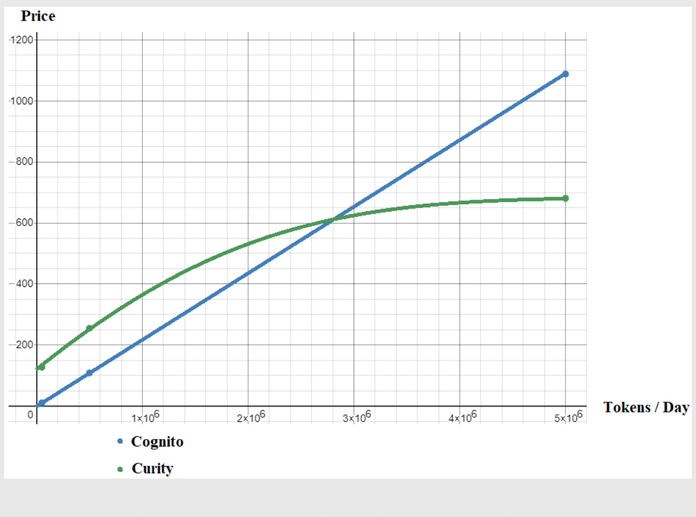
The thesis also presents differences in flexibility features, handling of use cases and security risks that need to be considered when deploying serverless applications.

OAuth 2.0 & OIDC Capabilites

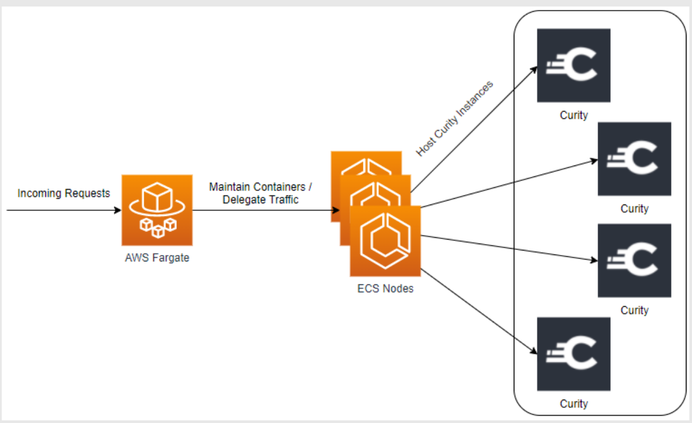
Supported Flows	Curity	Cognito
Code Grant Flow	Yes	Yes
Implicit Flow	Yes	Yes
Introspection Flow	Yes	No
Hybrid Flow	Yes	No
Client Credentials Flow	Yes	Yes
Refresh Token Flow	Yes	Yes
Token Revocation Flow	Yes	*
Resource Owner Password Flow	Yes	No
On-Behalf-Of Flow	Yes	No



Authorizing a standard client



Price scaling for Cognito (Blue) & Curity (Green)



Curity-deploymend on ECS Cluster

Conclusion

The results show that Cognito paired with other security services in AWS is enough for most common scenarios, however Curity is advantageous when handling large systems, systems with sensitive data (GDPR), systems that require high flexibility and micromanagement of OAuth behavior, and systems that require OAuth features not present in Cognito.