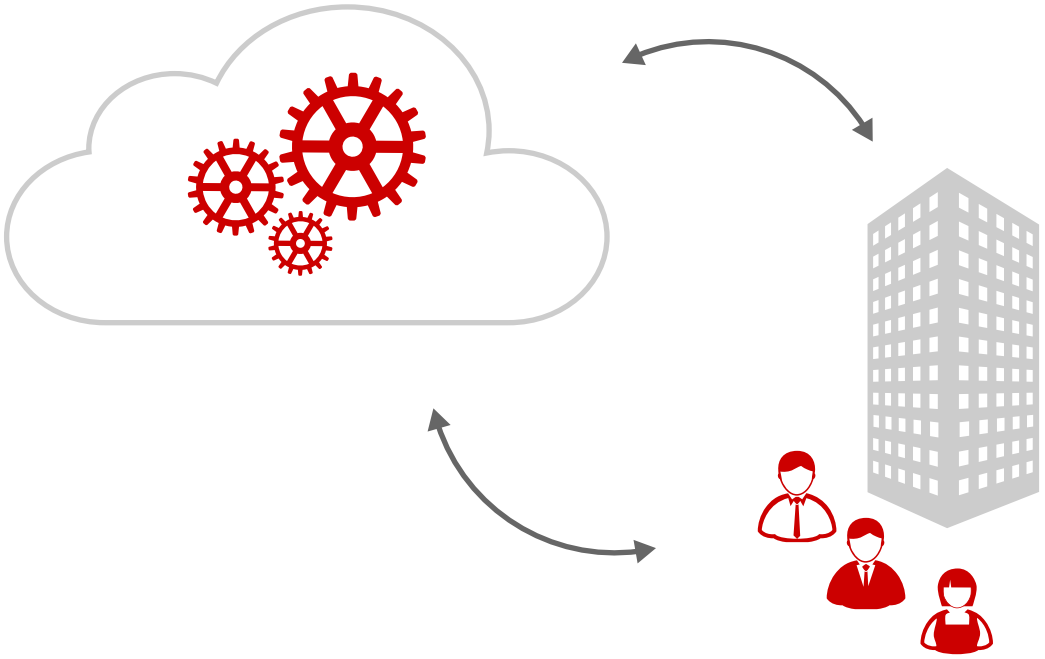
WHITE PAPER

# Cloud-based Enterprise Identity Management using OAuth



FORUM SYSTEMS™

THE LEADER IN API AND CLOUD GATEWAY TECHNOLOGY

# Cloud-based Enterprise Identity Management using OAuth

**SUMMARY:**

In this white paper, you will learn:

- The history of Authentication and Authorization.
- How OAuth standard is commonly used today.
- How enterprises are using OAuth and the benefits of using an API gateway in your company's architecture.

## OAuth Overview

The basic model of authentication and authorization over the internet is based on the traditional client-server model. In this model, at a minimum, there are two entities involved: the client and the application running on the server. A client with valid credentials is granted access to a particular resource controlled by the application. The client credentials may be in the form of a username/password that the application validates before granting access to the resource.

This basic model of authentication has evolved overtime as a result of the need for the client to provide its credentials (e.g. username/password) only once in order to be granted access to resources that are controlled by multiple applications in a distributed environment. This model is often referred to as Single Sign-On (SSO). In this model, the client "logs in" only once by providing its credentials to a single application. Upon validation by the application, the client receives a ticket (cookie) that enables it to seamlessly access resources of other applications. An example of SSO is a user logging into Amazon.com only once and accessing resources on multiple third party applications without having to login to each individual application.

The increased popularity of social media apps, mobile apps and cloud services has lead to another authentication and authorization model. The new model is based on the OAuth standard. In this model, at least three entities are involved: the user, the client application and the service provider. This is referred to as the three-legged OAuth model. The user is the owner of the resource and it grants client application access to its resources that are controlled by the service provider. OAuth standard enables the user to grant client application/service access to its resources without ever sharing its username/password with the client application.

## A simple example of OAuth

Traditionally, social media applications have been the main drivers behind OAuth deployment. In the past, web applications such as news media sites would maintain their own user profile data by providing the option to each of its users to create custom profile on the site for better user experience. This approach had many shortcomings for both users and media sites:

- Users had to provide email or username and password for each site during the initial creation of a profile
- Users would often forget their passwords during the login process since they had created multiple profiles at different sites.
- Media sites were now responsible for securing their users' emails and passwords from hackers.
- Users would often create fake profiles that would provide inaccurate tracking data to media sites.

Over time, social media sites such as Facebook, Twitter, LinkedIn, and Google have become the defacto repositories of a user's social identity or profile. The availability of existing social identities with rich profile data provided an opportunity for news media sites to access user data outside their domain of control. OAuth is the standard that enables websites to access user profile data outside their domain of control without requiring users to pass their username, password to the site.

Figure 1 illustrates a simple example that leverages OAuth. To create a better user experience for their visitors, websites (client application) provide the ability for users (resource owner) to post comments on articles with their facebook account. This ability allows Facebook profile attributes such as: name, photo and location to be displayed when they post a comment.

In order for this function to work, the user must give permission to the news media site to fetch his/her facebook profile information from facebook (service provider) without ever revealing his/her email and password to the news media site.

This popular example of OAuth creates a better user experience for website visitors of news and media websites and it reduces risk for the website owner. By using the visitors' facebook account, the website doesn't have to worry about storing account information of their website visitors/subscribers.

*FIGURE 1*

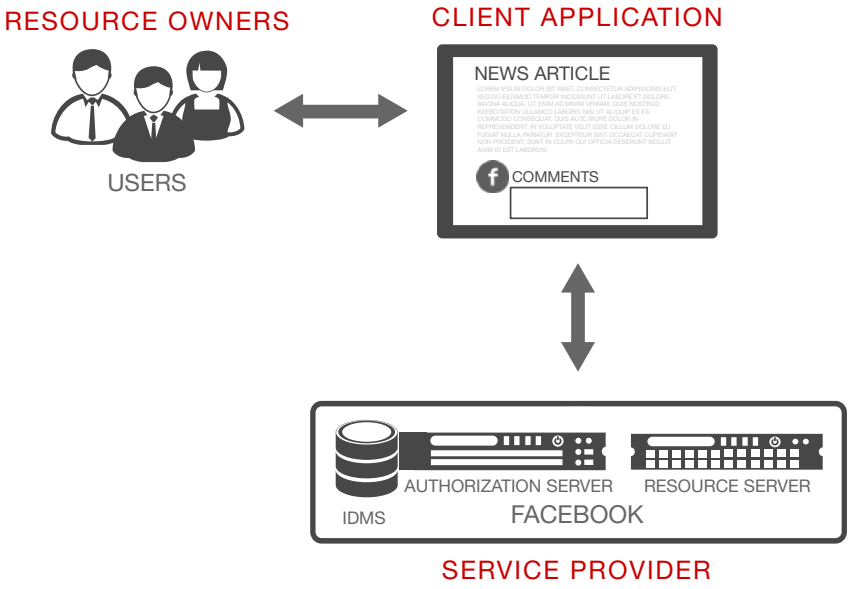



*Figure 1 shows a common use case where a user (resource owner) is attempting to post a comment on a new media site (client application) with his/her Facebook account. Before the comment can be posted, the news media site fetches the user's facebook profile attributes (user owned resources) from facebook (service provider). This is made possible by the OAuth standard.*

## Using OAuth for Enterprise Identity Management

The power and flexibility of OAuth in the social media sector has given enterprise companies an impetus to start adopting the OAuth standard for their cloud-based enterprise identity management. A prime example of this adoption is based on a use case where a company's email system is hosted in the Google cloud. Google cloud is the identity repository for the company's users. The company has all its registered users validate their emails and passwords with Google cloud before being allowed access to company applications.

Figure 2 illustrates an architecture deployment of a company leveraging a cloud based identity management system to control access to its company applications.

Although, a cloud-based access control architecture may appear to be straightforward and simple, it can certainly pose several challenges for an organization:

- Company applications require modification to be OAuth enabled.
- Over time, scalability becomes an issue. As new applications are deployed, they must be integrated and tested with OAuth, which requires time and resources.
- This deployment doesn't offer any centralized monitoring and enforcement.
- Performance becomes an issue when SSL is used by applications to exchange OAuth credentials with Google cloud.

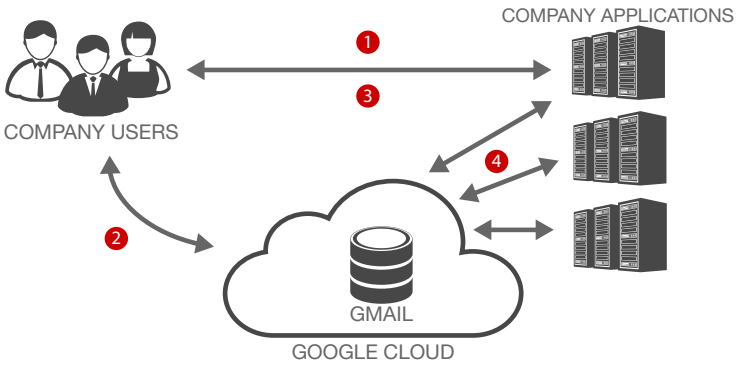
## Using OAuth with an API Gateway

When you add an API gateway to the architecture deployment, it solves many of the challenges previously mentioned:

- No modifications required to company applications. Applications are OAuth agnostic.
- Scalability is no longer an issue as new applications are deployed. Integration and testing of OAuth is no longer required with applications.
- Centralized monitoring and enforcement is easier with an API Gateway. API Gateway provides full visibility to who is accessing what resource.
- Performance is no longer an issue since an API gateway accelerates SSL traffic that contains OAuth credentials.

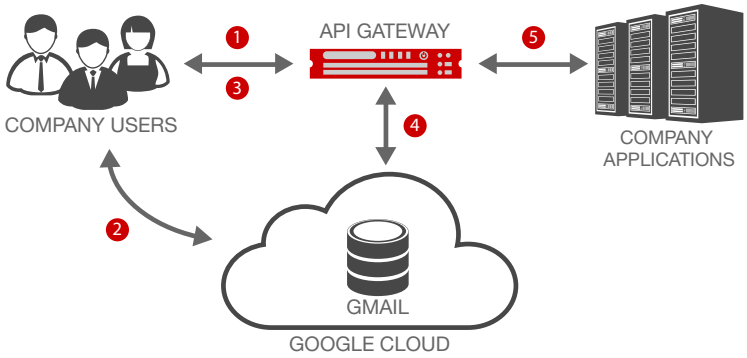
Figure 3 shows how the deployment of an API gateway changes the process of accessing the cloud-based identity management system.

*FIGURE 2*




1. User attempts to access company's applications.
2. User is redirected to Google cloud to provide company email & password for authentication.
3. Upon successful authentication to Google cloud, user is redirected back to company applications with OAuth credentials.
4. Company applications that receive the user's OAuth credentials validate with Google cloud before granting access to application services.

*FIGURE 3*




1. Users attempts to access company's applications.
2. User is redirected to Google cloud to provide company email & password for authentication.
3. Upon successful authentication to Google cloud, user is redirected back to API gateway with OAuth credentials.
4. API gateway validates user's OAuth credentials with Google cloud.
5. Upon successful validation, user is granted access to company's application services.

When it comes to deciding whether your company needs an API gateway in your architecture, there are several factors to consider. It's important to evaluate how many applications are needed to achieve your business goals. If you have only one application and don't have plans to add more in the future, you probably don't need a gateway. However, if you are adding new applications services based on new business requirements, deploying an API gateway will save you a lot of time and resources while providing a more scalable and modular architecture.