Whitepaper

Evolve your Authorization Strategy with Fine Grained Authorization

Introduction

Use of SaaS applications is more widespread than ever — with consumers, employees and professionals of all types logging in and out of multiple apps daily to do everything from collaborating on a work project, to checking on test results after a doctor visit, to accessing mobile banking resources. Highly collaborative apps like Canva, Notion and Figma in particular have seen rapid rates of adoption, putting pressure on legacy SaaS providers to add more innovative features to their own offerings.

But as SaaS applications grow increasingly sophisticated, collaborative, and feature-rich, authorization gets more complex. Traditional methods for securing and managing access to documents, files and other types of assets — first generation point solutions and in-application DIY coding — simply aren't flexible or scalable enough. Today's apps need more precise access control.

This is why Okta is introducing Fine-Grained Authorization (FGA) — a new approach to managing access control that's not only more secure, but also easier to implement and scale.

More collaborative software means more complex authorization

Businesses want to be more agile and efficient, and have been pushing SaaS developers to offer more sophisticated collaboration features in their products. This demand is being driven by a need to better serve both external customers as well as employees.

According to Corel's recent Collaboration survey, employees know they can do their job better with collaboration tools, but their workplace apps are falling short. 82% agreed that poor collaboration is limiting their productivity and wasting their time, and 62% agreed that poor collaboration tools were impacting overall business growth.

Authentication vs. Authorization

Authentication allows you to verify who the user is. Authorization is about verifying if that same user has the right level of access within an application.

As the collaborative capabilities of SaaS solutions grow, however, so do authorization headaches. Suddenly there are exponentially more decisions around who gets access to which assets. What level of access will each user have? How will permissions be handled for external vs. internal users? How quickly do permissions need to be turned on and off? In this landscape, more granular and specific authorization is essential.

For example: a single document may have multiple levels of permissions (e.g. admin, editor, commenter, view-only) for multiple users, according to a host of parameters (e.g. such as role, location, title, etc.). SaaS developers need to figure out how to provide the right level of access for a growing number of applications and assets. And do it for a user base that may span employees, partners, customers, contractors and more. This compounds the complexity of securing and managing authorization.

More points of access = more security and compliance vulnerabilities

Why do more sophisticated collaboration capabilities present more vulnerabilities? The answer is pretty simple — when there are more points of access to critical documents, files, content and data within an application, there are more points of risk. In fact, “Broken Authorization” is the #1 threat on OWASP’s Top Ten API Security Risks. This is because of the detailed logic in user policies and hierarchies.

Today’s SaaS apps also need to meet compliance and auditing rules, especially in regulated industries like financial services, legal, and healthcare. Many companies need to both share critical files and documents, and also keep on top of which users have permission to access which assets for reporting and auditing. As with security vulnerabilities, compliance exposure increases as the user base, products, and features of a SaaS app multiply.

The need for more granular authorization is clear, but the solution is easier said than done. In the past, developers implemented authorization using traditional methods, often DIY, primarily based on roles that can’t adapt to complex use cases. Developers need new solutions to mitigate this challenge.

FGA vs. traditional methods

Currently, most businesses use role-based access control (RBAC) for authorization. It is a coarse-grained approach that grants permissions based on predefined roles. For example, RBAC can provide access to a resource, like Jira, to an employee with a particular role, such as a member of the product marketing team. But as SaaS apps add more and more features, this model is quickly becoming outdated. It is not granular enough, and not secure or compliant enough, for the long haul.

Administrators or team leaders may need to grant differing levels of access to projects, assets, and capabilities according to a range of parameters, not just title or role. Often, it is a dynamic combination of attributes, as well as least privilege access principles, that factor into who should have access to an asset and who shouldn't. Here are some examples:

- A recruiting company wants to give permissions to the hiring manager to review all candidates for a position; meanwhile, each interviewer should only be able to access details for the specific candidates they are interviewing.
- A shared family banking app needs to provide varying levels of permissions to each account holder — including spouses who share an account, as well as two teens who have limits on amounts that can be withdrawn or transferred without parental permission.
- A company wants to give a support engineer access to customer data for troubleshooting purposes, but only for a limited time, while a support ticket for that customer is open.
- An SaaS provider needs to give its developers different permissions on different cloud servers. (e.g., a "development" service can be accessed by every developer, but the "production" service is able to be accessed by only a few.)

Least Privelege Access

Least privilege access is the practice of only granting access to resources and assets that a user must have to perform a specific activity. This is a security best practice designed to mitigate risks associated with application access in an increasingly complex technology landscape that encompasses remote work, cloud services and more.

The 2021 Global Cybersecurity Survey Report indicates that two out of three organizations now consider least privilege a top or urgent priority. Among those that failed at least privilege access efforts, 32% indicated complexity as the main reason their least privilege strategy failed.

So how are businesses handling this challenge now? Development teams often need to piece something together using a combination of point solutions and custom coding. But this "duct tape" approach only causes additional challenges, including the following:

- Too complex to manage. Addressing authorization at the application level makes it too complex and time-consuming to manage. And it's very difficult to scale as new applications, and corresponding features and permissions are added. Businesses need a centralized view of authorization.
- Compliance and audit gaps. Businesses need to efficiently audit which users have access to which assets in order to meet regulatory and compliance rules. Without a centralized view into authorization policies, businesses cannot achieve their compliance goals.
- Multiplying security vulnerabilities. Authorization implemented within application code can create inconsistencies that increase the attack surface of your SaaS offerings. The greater the attack surface, the greater the risk of a security breach.
- Stress on development resources. Retrofitting traditional approaches to solve current authorization challenges doesn't solve the problem. Developers will spend too much time with homegrown systems and building authorization capabilities into application code over and over again. That's time that could instead be spent on product innovation that helps you stay ahead of the competition.

What's Needed to Make Authorization Work in a Modern SaaS Landscape?

Today's SaaS solutions need more granular authorization options, but DIY attempts at addressing this issue at the application level are not practical, scalable or secure enough. There are several key elements that are needed to make authorization work in the long term:

- **Centralized.** Developers need a means to take authorization out of the application code and implement access control and permissions centrally, across the SaaS landscape.
- **Flexible.** Authorization capabilities need to go beyond RBAC and be flexible to address the varied and constantly evolving access requirements of the SaaS user base.
- **Scalable.** Re-creating the authorization wheel every time a new application comes online is inefficient and keeps development teams from shipping new innovation quickly. Authorization solutions must be easy to deploy, adjust and scale as needs change over time.
- **Secure and compliant.** Authorization logic that is difficult to audit and manage leads to security and compliance risks. The goal should always be to reduce points of vulnerability, not add to them.

What you should look for in a modern authorization solution:

- ☐ **Low latency** — every millisecond you spend in authorization will impact your application's overall latency.
- ☐ **Availability and reliability** — authorization is mission critical and any downtime can disrupt user access in your application.
- ☐ **Scalability** — your authorization solution should be capable of supporting your businesses growth now and in the future, planned or unplanned.
- ☐ **Policy flexibility** — support for different kinds of policies like RBAC or ABAC give you the ability to choose the best policy for your app; and update policies as your needs evolve.

Introducing Okta Fine Grained Authorization

Businesses and developers need flexible solutions for authorization, whether that's coarse-grained, fine-grained or anything in between. They need solutions that are reliable, compliant and secure. That's why we built Okta Fine Grained Authorization.

Okta Fine Grained Authorization (FGA) delivers authorization at scale and gives businesses the power to simplify access control across multiple applications, parameters, and users. It's **authorization as a service**, allowing developers to design and implement permissions in a way that's flexible, scalable, and easy to use.

- **Manage authorization at scale, without complexity.** With Okta FGA, developers can update and manage authorization policies from one centralized location without touching application code. Scale access controls as your product and user base grows.
- **Get as granular as needed, with absolute control.** Easily manage groups, teams, organizations, or any set of users, and assign them permissions on any resource or groups of resources. Developers can define access down to the finest detail, ensuring greater security and compliance.
- **Be flexible with permissions and access.** With Okta FGA, app owners and developers have greater flexibility in defining how users grant permissions and access. Their users can then create authorization rules around multiple parameters, beyond just roles.
- **Reduce Latency.** Okta FGA is designed to provide high scalability while minimizing latency, by routing requests to the closest server and responding to authorization queries very quickly. Users can move at the speed of their business.
- **Save development time and resources.** Okta FGA saves development time and resources by making it easier to build and scale software with sophisticated authorization capabilities built in. It seamlessly integrates with a business' existing systems using developer-friendly tools like APIs, SDKs, CLIs and IDE integrations.
- **Backed by the Okta brand.** We pioneered trusted authorization, now we're pioneering FGA.

Further Reading

To learn more about Okta FGA, please see:

FGA.dev

Getting Unlimited Scalability to Okta FGA

Supercharge your Authorization System with FGA

Conclusion

As the collaborative features of SaaS offerings grow in complexity, FGA needs to be part of your security and compliance strategy. With FGA, your developer team will spend less time maintaining authorization and more time on innovation.

Does your developer team need FGA?

If you answer yes to one or more of the questions below, then you need FGA.

- [ ] Are you adding or **enhancing collaboration features** for your customers?
- [ ] Do you sell to businesses in highly regulated industries with extensive **auditing and compliance requirements**?
- [ ] Do your customers **need more granular control** than what RBAC offers?
- [ ] Is your access control decentralized, requiring **constant updates and coding** within multiple apps?
- [ ] Are your **developers spending too much time managing authorization** instead of on product innovation?

Next Steps

Learn more about Okta FGA in our Playground

Get started with Okta FGA