State of Authorization: Playbook Edition

Foreword from CTO

There is growing, broad acceptance that authorization is no longer a 'nice to have,' but is an integral component of a modern cybersecurity strategy. This is due to an ever-increasing amount of services, data, users, and interactions that require that the right level of access control be applied as well as increased legislation businesses need to comply with.

But a challenge remains - as enterprises consider implementing an external solution to authorization, they often tell us that it's too difficult, will take too much time, and that they do not have resources with the skill depth or time to effectively implement an authorization solution.

When we published the first State of Authorization Report, our goal was simple – address a few core issues that continued to pop up in discussions with customers, prospects and partners worldwide.

We release the State of Authorization: Playbook Edition as a guide that addresses every step of the authorization journey, from ideation through deployment to help organizations succeed with their deployment, better understand common challenges associated with authorization and get answers to popular questions about choosing the right path forward to implement authorization. We look at topics you might be grappling with now including whether to build or buy, the differences between authentication and authorization, how authorization can optimize existing IAM investments, as well as issues that may pop up in the future, including how best to leverage Generative AI with authorization.

The authorization landscape is thriving with new vendors, frameworks, and more importantly accrued activity in the standards communities (IETF's OAuth, OpenID Foundation, and the Cloud Native Computing Foundation). There is no better time to get on board with the authorization journey.

As Axiomatics, we've witnessed first hand and contributed to the authorization landscape since the beginning. For over 15 years, we have been thought leaders in the authorization space and draw from our experience implementing dynamic authorization solutions for customers large and small in a wide range of industries

I'm proud of the content we're delivering in this playbook. I hope you find our State of Authorization: Playbook Edition informative and insightful guide to all things authorization.

David Brossard
Chief Technology Officer, Axiomatics

Table of contents

= NEW

Glossary

A to Z of authorization

Acronym	Defnition
ABAC	Attribute-based Access Control
AI	Artificial Intelligence
ALFA	Abbreviated Language for Authorization
API	Application Programming Interface
CI/CD	Continuous Integration/Continuous Deployment
CIAM	Customer Identity Access Management
FGAC	Fine-grained Access Control
IAM	Identity Access Management
IGA	Identity Governance and Administration
NIST	National Institute of Standards and Technology
OASIS	Organization for the Advancement of Structured Information Standards
OPA	Open Policy Agent
PAP	Policy Administration Point
PBAC	Policy-based Access Control
PDP	Policy Decision Point
PEP	Policy Enforcement Point
PII	Policy Identifiable Information
PIP	Policy Information Point
RBAC	Role-based Access Control
SoD	Segregation of Duty
XACML	eXtensible Access Control Markup Language
XML	Extensible Markup Language

What is authorization and how is it different from authentication?

Authentication and authorization are two very important and closely related yet different parts of the information security process for organizations. Put simply, authentication is the process of ensuring that a user is who they claim to be. Authorization is the process of ensuring that once an authenticated user is in the system, they are only permitted to perform actions that they are allowed (or authorized) to do on the right resources. Authentication and authorization are meant to be used together, with authentication happening first, followed by authorization.

The more technical view is authentication is the process of verifying the identity of the user or the claim the user is making (such as their age) and ensuring that the user is who they claim they are. The challenge with authentication is that it is entirely identity-centric and focuses on validating the trustworthiness of a claim, not its subsequent use in the underlying system. Authentication doesn't care what the user is here to do.

Authorization is the process of ensuring users have access to resources and allowing them to perform relevant actions on the resources - but only to the extent allowed by policies and constraints imposed by the organization's business, legal or any other categories of requirements.

Why is it important to get authorization done right?

A properly implemented authorization model protects the crown jewels of the business and ensures only authorized users have access to view, edit, create or perform any related actions on them.

An improperly implemented authorization model can lead to leaks of sensitive data, breakdowns of integrity around critical processes, brand damage, customer disruption, and in extreme cases - legal penalties.

How does externalized authorization come into play?

Externalized authorization refers to the process of performing and managing authorization outside of or separate from the application being protected. Instead of delivering a single application that implements all required features, the software developer simply implements core business functionality and reuses common blocks for non-functional aspects such as authentication, logging, and data storage.

Attribute-based access control (ABAC) enables externalized authorization by leveraging the principle that software code should be decoupled based on the function it serves. In essence, having authorization decoupled from the application reduces the need to touch application code every time there is a business, regulatory or internal change. Plus, it lets developers stay focused on building great applications, not security processes.

It is good to note that ABAC did not invent externalized authorization - the concept of externalized authorization predates ABAC.

Key Takeaways

- Authentication is the process of ensuring that a user is who they claim to be.
- Authorization is the process of ensuring that once an authenticated user is in the system only allowed actions are in fact being performed.
- Authentication and authorization work together to improve security by making sure only authorized users are allowed access to resources and what actions they can perform with that access.

Why attributes?

Historically, many organizations have tried to shoehorn everyone into a role, group, or title. But the truth is that the world is more complex. There are more parameters to take into account during the authorization process.

These parameters can be expressed as attributes, which are labels - or pieces of information. Attributes can be about anything: the user, the resource being accessed, the action being attempted, and even contextual data such as time of day. In their simplest form, an attribute is a key-value pair e.g. role=="manager" or department=="sales". Attributes should also come in sets of an identifier (or key) and the value or values identified with it.

For example, consider a user's name is John Smith, which is the identifier. Some of their attributes are role, citizenship, risk level, office location and so on, which are the values.

Extending beyond roles with attributes

Role-based access control (RBAC) is a model that expresses the notion of access in terms of users, roles and entitlements (permissions). A role is simply a way to distinguish which users should have authorization to access certain information. A role maps nicely into an enterprise hierarchy as well: you are hired as an office manager, a software developer, or an intern. Users fits naturally into those roles.

A simple explanation of RBAC is that a user can have one or more roles. These roles may mirror their business roles or they could also be simply a way to represent a business function being performed in some abstract form

Roles are limited in what they can express and that leads to scalability issues. Many organizations end up with a 10-to-1 role-to-employee ratio and this leads to the infamous role explosion. As the number of roles increases, the process of creating and updating roles becomes more complicated and error-prone, increasing the risk of misconfigurations or other security issues

The coarse-grained limits of a RBAC approach can also increase the risk of exposure. As RBAC is only supported by a static authorization methodology that can't expand access beyond a role alone. Not to mention, it is also not content-aware so it can't cater to time, location or risk.

ABAC is the fine-grained and context-aware approach to authorization compared to RBAC. The ABAC model extends your RBAC strategy by considering other attributes in addition to roles. ABAC relies on the various attributes of entities involved in the process to express the authorization policies and enforce access control requests

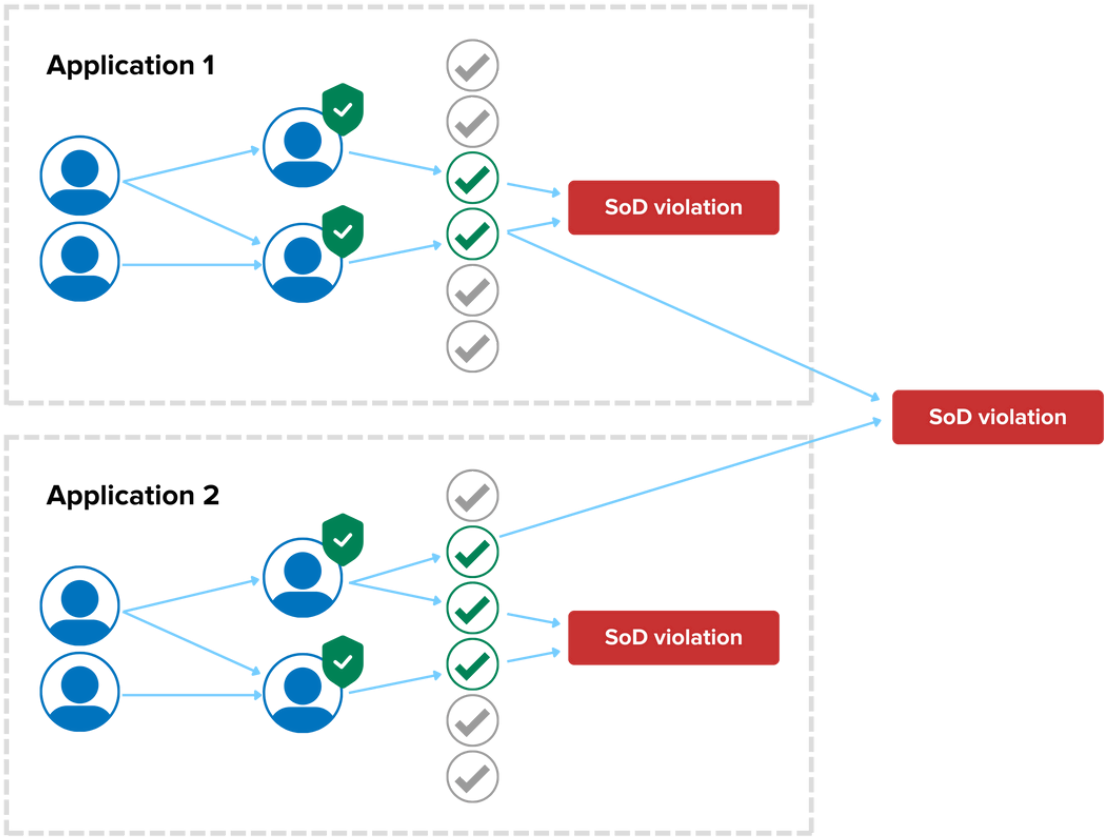
The fundamental change in an ABAC approach compared to RBAC is to recognize that multiple attributes of users, systems and even the operational environment would need to be considered when deciding whether or not to allow access. ABAC takes into consideration what you represent, what you want to do, what you want to access, for which purpose, when, where, how, and why instead of only who you are.

Considering entitlements

An entitlement represents a record which grants access to an information system. A true entitlement is a record of a security assignment made by a human decision, but there are numerous definitions of what an entitlement is, each with small differences between them.

However, managing entitlements continues to cause challenges for enterprises. The issue is related to limitations - an entitlement is a very specific identity-centric element in an overall access control strategy. Their over- or under-use can provide a myriad of issues for an enterprise, opening them up to risk.

It is called never-ending sudoku because while removing conflicting permissions from role 1 and/or role 2 may solve the problem for user group 2, it creates new problems for user group 1 and 3. This creates a back and forth with changing the permissions until no issues arise.



However, ABAC has the power to capture the complete implications of organizational access control policy in the form of rules (policies) and attributes. Attributes may be objective properties (for example that a user has received training) or entitlements (granted by management decision).

What's the difference between ABAC and PBAC?

Over the past few years, some vendors have started using the term Policy-based Access Control (PBAC) as a way to differentiate the messaging around their access control product offerings.

PBAC and ABAC are essentially interchangeable in that they enforce policies using attributes. The policies are made up of rules, which are expressed in terms of attributes and their values.

The key difference between them is which "end" of the access control model stack you look at: policies that inform the authorization engine what to do and attributes which inform the authorization engine how to do it.

Key Takeaways

- RBAC expresses the notion of access in terms of users, roles and entitlements (permissions).
- RBAC is only supported by a static authorization methodology that can't expand access beyond a role alone.
- ABAC gained traction as an access control model that extends your RBAC strategy by considering other attributes in addition to roles.
- Entitlements, like roles, are limited in what they can do as they are a very specific element in an overall access control strategy.
- The over- or under-use of entitlements can provide a myriad of issues for an enterprise, opening them up to risk.
- PBAC and ABAC are essentially interchangeable in that they enforce policies using attributes.

Policies / Policy Lifecycle Management

Attributes alone are not enough to express authorization logic. Policies link roles with other attributes to provide a rich palette of access control possibilities. A solid access control strategy uses policies to ensure that the right data or processes are accessed at the right time by the right people.

Taking this into consideration, many organizations wonder how to build policies that not only reflect appropriate access, but also address all applications and databases (whether in the cloud or on-premises) while being both repeatable as well as scalable.

Policy languages

In order to create policies we must understand how they are written. There are many policy languages out there, but these are the main four.

eXtensible Access Control Markup Language (XACML)

XACML is a standard, which was developed by leading security experts under the umbrella of OASIS (organization for the advancement of structured information standards). The language defines three key elements that enable fine-grained authorization to be implemented across different deployment models such as cloud, on-premises, and hosted environments: a policy language, a request/response scheme, and an architecture.

Abbreviated Language for Authorization (ALFA)

ALFA is a domain-specific language used to express fine-grained authorization policies in a lightweight and developer-friendly syntax. It borrows the same authorization architecture from XACML and NIST's Guide to Attribute Based Access Control (NIST SP 800-162). ALFA also borrows its structure and hierarchical nature from XACML making it easier to manage and grow the number of policiesALFA is a domain-specific language used to express fine-grained authorization policies in a lightweight and developer-friendly syntax. It borrows the same authorization architecture from XACML and NIST's Guide to Attribute Based Access Control (NIST SP 800-162). ALFA also borrows its structure and hierarchical nature from XACML making it easier to manage and grow the number of policies

ALFA enables 'policy-as-code' as developers can now write policies by hand in their favorite IDE and manage the policy lifecycle through their regular CI/CD processes. This tremendously reduces the time needed to implement a comprehensive policy-driven authorization framework.

ALFA can be used to configure XACML-based authorization engines (also known as PDP) Together with the JSON and REST Profiles of XACML, ALFA makes the adoption of externalized authorization much easier and faster on developers.

AWS Cedar

Cedar is a policy language and evaluation engine developed by AWS which enables developers to express fine-grained permissions. The permissions are expressed as policies that are enforced in but decoupled from the applications they protect.

Open Policy Agent (OPA) and Rego

Built on Rego, OPA is a general-purpose policy engine which unifies policy enforcement across the stack. It offers community-created connectors for attributes and endpoints that developers can use to drive effective authorization within your organization.

Creating policies

When creating policies there are many things to consider including whether the policy author will write policies by hand or have a tool to do it, identifying who is responsible for creating the policies and their level of expertise and familiarity with policy creation.

When you pick a policy language, it is important to consider the completeness of the language and how it will address your security requirements, auditing and logging, integration capabilities as well as ease of use. For example, some languages like Rego and OPA are harder to audit whereas ALFA is easier to write, manage, test and audit. It can also be easier to use a tool to write the policies rather than creating them by hand, which in many organizations leads to questions around who should write the policies

Regardless of who writes the policies, many people across the organization will be involved in the policy creation process at various stages of the process:

1. **The application owner is responsible for defining the overall use case.**
2. **Business analysts, security architects, and security officers (compliance & privacy managers) are responsible for defining authorization requirements.**
3. **Business analysts, architects, and data owners are responsible for defining the required attributes and their source.**
4. **Application developers or policy authors are responsible for authoring policies.**
5. **Application developers, policy authors, and business analysts are responsible for defining, implementing, and running policy tests.**
6. **Architects and application owners are responsible for deploying the architecture and the policies.**
7. **Compliance & audit managers and application owners are responsible for running ABAC access reviews.**

When it's time to write authorization requirements and the subsequent policies, business analysts and policy authors should strive to think in plain and simple terms. Those used to an intensive role-engineering process should set that baggage aside and go back to the basics of what should happen. It is sometimes helpful to think in terms of the who, what, when, where, why, and how (the 5W and 1H).

Another thing to consider when creating policies is versioning. This can't easily be done with RBAC, but ABAC enables you to go back in time and edit policies.

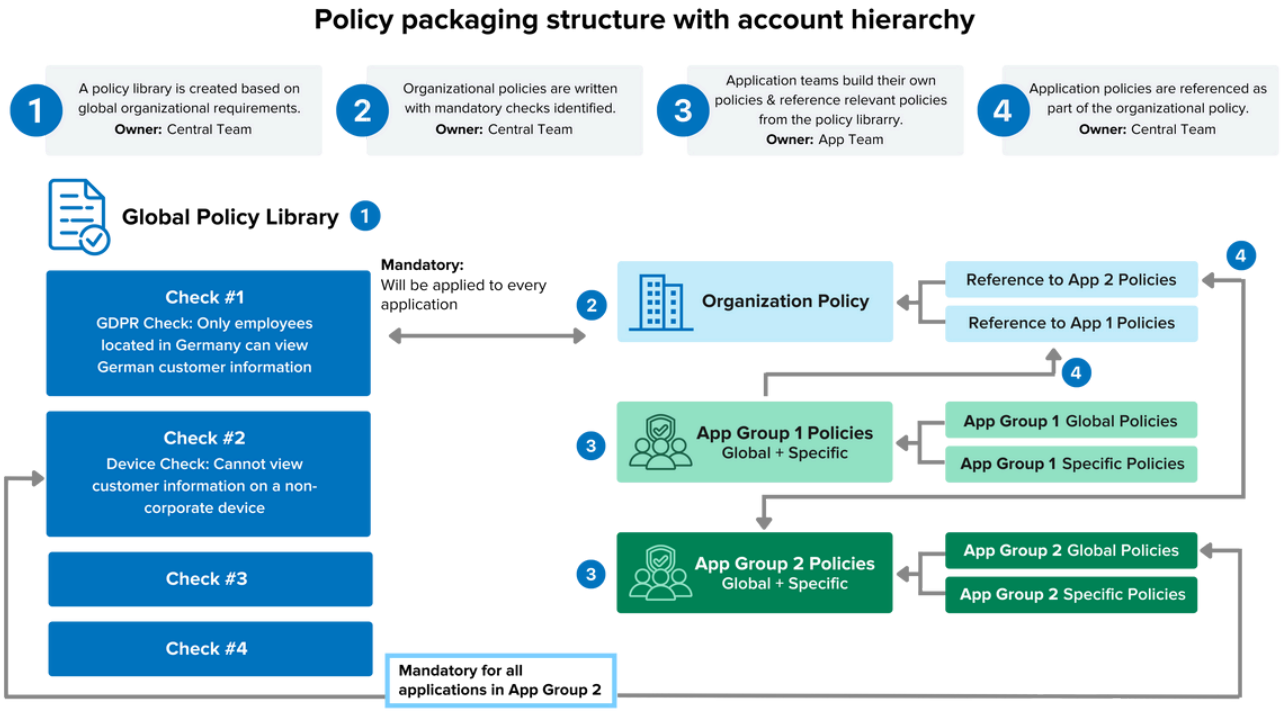
Policy-based authorization also delivers two additional benefits:

1. With regards to versioning, it is easy to keep track of subsequent policies and even roll them back to a previous state. In previous authorization models such as RBAC, it's nearly impossible to keep track of the evolution of the authorization logic as it depends both on roles & groups and their interpretation in the application's logic.

2. With regard to authorization conflicts, accumulations and contradictions, as well as segregation of duty (SoD), policy-based authorization (and in particular ALFA and XACML) provides us with natural tools to address conflicts. Specifically, policy authors can decide which policies override others (either expanding or restricting access). SoD rules such as not being able to approve a payment an individual has issued themselves is straightforward in policies while next to impossible with an RBAC approach.

Enterprise policy structure NEW

An enterprise policy structure is a framework where policies are organized hierarchically, usually with overarching global policies at the top, followed by organizational policies, and then specific policies for individual applications or business units. This structure provides a clear delineation of authority and control while enabling efficient management of access policies across the enterprise.



The example above shows a policy packaging structure that considers account hierarchy. It is important to note that this structure offers endless flexibility to balance both central control and decentralized authoring to achieve both the security and business needs of the organization.

There are many benefits enterprises can realize when using this approach, including:

Enhanced control and governance

By establishing global policies at the top of the hierarchy, enterprises ensure consistency, as well as the ability to adhere to overarching security standards across all business units and applications. This centralized control mitigates the risk of unauthorized access and ensures compliance with regulatory requirements.

Flexibility and adaptability

The structure enables enterprises to tailor access policies to specific business needs and requirements. As policies can be created and managed at different levels of the hierarchy, internal stakeholders can easily adapt to evolving security landscapes, organizational changes, or new business initiatives.

This agility is crucial to ensure your security and identity teams can respond quickly to emerging threats or opportunities.

Scalability and efficient collaboration
By organizing policies into logical building blocks and leveraging a common taxonomy, enterprises can streamline policy creation, maintenance, and enforcement. This not only improves operational efficiency but also reduces the risk of errors or inconsistencies, which pop up in a flat policy structure.

Clear separation of concerns
The structure enables organizations to create a clear separation of concerns, allowing different teams or business units to manage their own policies, but within the guardrails set by a central team. This decentralized approach empowers teams to make policy decisions that align with specific requirements and objectives while still adhering to overarching security guidelines set by central governance bodies.

Policy creation lifecycle

The policy creation lifecycle is the journey the teams will go on each time a policy is created. This helps ensure each step is taken, so that the end result is meaningful authorization for the organization that reflects the original business requirements.

Step 1 - Define the use case
The first step in the policy creation journey is to discuss and frame the use case with various stakeholders. The stakeholders could include the IT team, business analyst, developers, application owner and security architect.

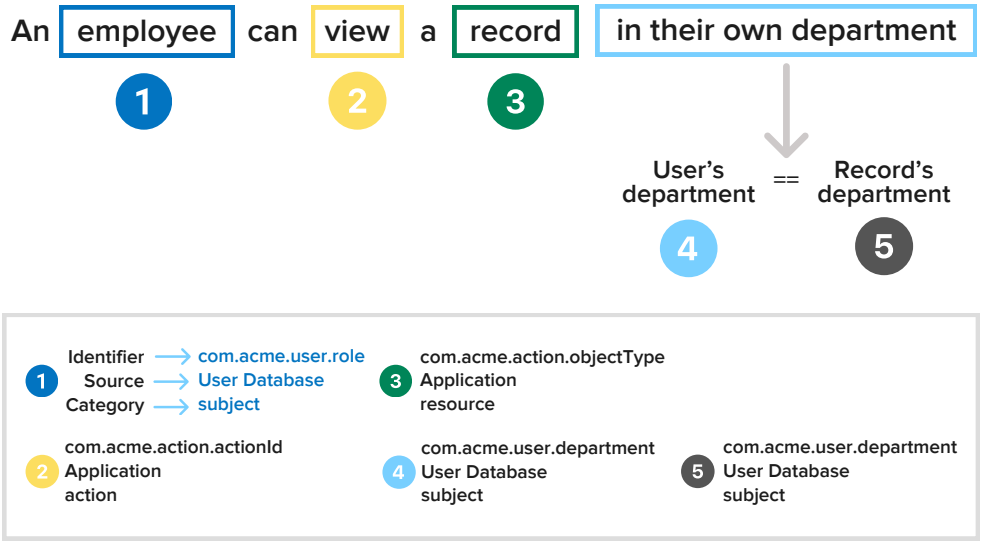
When choosing a use case the organization wants to ensure to pick a well-scoped case to guarantee success. An example of a use case is an organization wants to let its users access records easily. This could apply to employees, customers, and partners, but not all individuals should have access to all records

The goal of this step is to provide context and achievable scope of the project.

Step 2 - Gather authorization requirements
Once the use case is defined, the next step is to identify all of the requirements, regulations, and other rules. If the requirements/regulations are not defined, now is the time to find or create them.

Authorization requirements may come from several sources and stakeholders. For instance some requirements may relate to how a business operates (e.g. working hours). Others may relate to security best practices (e.g. encrypting data). Others still come from regulations (e.g. privacy regulations or PCI-DSS). Others come from the business owners.

When forming the authorization requirements they should be in the following format: Subject + Verb + Object + Context. An example of this in use is:



When forming the authorization requirements it is important to not do a role engineering exercise - writing policies requires a different thought process when compared to roles.

The goal of this step is to gather all the requirements and identify their owner (or stakeholder).

Step 3 - Identify required attributes

Once requirements have been listed, it now becomes necessary to identify the attributes used in those requirements. In this stage, read through the natural language statements previously defined and break them down into attributes (it's time to pull out the highlighter!). Identify each one of them and ideally identify their source. The more specific the definition, the better the policy will be and the likely policies and requirements will be clearly understood. Lastly, this approach will facilitate access reviews.

Step 4 - Author the authorization policies

When authoring policies it is important to implement a good policy structure which reflects your business. This also makes it easier for maintenance to be done in the future.

The diagram above summarizes the process through which natural language statements are broken into attribute-based policies. They can then be directly implemented into ALFA or XACML.

Step 5 - Test the policies

Once requirements, attributes, and machine-interpretable policies are defined, policy authors should agree on and implement acceptance tests. While defining acceptance tests is part of the authorization requirements step, this step focuses on implementation.

It is important to put the responsibility to define the tests in the hands of the application owners and the business analysts. Policy authors and application developers are merely responsible for implementing the tests into machine-executable tests.

There are different types of tests that can be run against ABAC policies:

Binary testing
Can a given user perform a given action on a given resource under a set of given circumstances?

Gap analysis
Is there any way a given user can perform a given action on a given resource?

Reverse query testing
What can a given user do? Who can access a given resource?

It is good to note that the different testing features are not done within the individual policy languages themselves. The testing is done within the different authorization or policy solutions and it is what differentiate products from one another.

Step 6 - Deploy the architecture

Once tested you can deploy the architecture. With ABAC, you can choose where and how to deploy your policy enforcement point (PEP). Where you deploy the PEP determines what type of authorization you can achieve and how broad the protection is.

The location of the PEP and the interfaces it goes against may impact access granularity. For instance, integrating with a web single sign-on (SSO) layer will be relatively coarse-grained as the PEP will not have access to any message payload. Integrating with an API gateway inside the API tier will be finer-grained as the PEP will have visibility into the API message on the way in and on the way back.

It is the responsibility of the application developer, application owner, and architect to define the most suitable place for a PEP. It is good to note that there can be multiple PEPs deployed at the same time.

Step 7 - Deploy the policies

When deploying policies you will want to do this in stages from development, testing, QA and finally production.

Policies can be handled as code would be handled. Policies can be checked out from a development system, committed to a version control system, tagged, and checked out in the next environment. Change and version management should be handled through those tools already in place within the enterprise for source version control. It is the responsibility of the application owner jointly with the owner of the centralized authorization ABAC service to deploy new versions of the policies.

In an ABAC system, the policy decision point (PDP) is a stateless authorization engine. This means the PDP can be deployed alongside other PDPs behind a load balancer.

An ABAC solution should have the means to push configuration to an entire set of PDPs in one go. An ABAC solution should also provide the means to rollback policies to a previous version.

Step 8 - Run access reviews

Lastly, you want to run access reviews to ensure who can do what and who did what. In an ABAC system, an access review consists of an analysis of the policies against a set of attributes to determine what these attributes grant. For instance, given a set of user attributes (identity, role, department), what can those users achieve?

Such reports can be produced by evaluating partial requests against an ABAC authorization engine configured with the relevant policies.

The key benefit of this approach is that it shows the direct root "cause" for access to be granted. For instance, given the access review request "what can a manager do?", the report will contain:

- View documents if user.department == doc.department
- Edit documents if user.userId == doc.owner
- Etc...

In the ABAC authorization policy lifecycle, it is important to run access reviews on a regular basis to determine whether a given user profile (i.e. a manager) or a specific user (i.e. Alice) have gained or lost entitlements.

Key Takeaways

- There are four main policy languages: XACML, ALFA, Cedar, and OPA (based on Rego).
- Evaluate language options based on flexibility of authoring, compatibility to interoperate with organizational resources and scalability to tackle demands of both security requirements as well as regulations and compliance.
- Policy authors will want to write the policies as simple English statements in the form "a user can (or cannot) do a given action on a given item" and move away from the thought process consistent with a RBAC approach.
- The authorization policy lifecycle aims to simplify the deployment and adoption project.

Build vs buy

When evaluating the best approach to securing your APIs, applications, databases, and cloud, organizations are often caught in the decision of building a framework in-house versus engaging a vendor to help or using an open-source framework.

There are several key advantages to partnering with a vendor that specializes in fine-grained access control solutions rather than trying to build an equivalent solution using in-house resources.

Budget

The costs associated with building and maintaining an ABAC solution may be one of the first and most logical reasons for a business to avoid choosing this option. Building a solution in-house may seem like a smart way to have full control over your budget, but in reality, unexpected costs are more likely to arise than if you went through a partner. Building a custom solution and maintaining it is almost always going to be more expensive.

It also implies having to design the solution from scratch whereas vendors and OS frameworks have already laid the groundwork.

According to a McKinsey survey of IT executives, large IT projects run over budget 45% of the time, while delivering 56% less value than planned. The same study found that 17% of projects go so badly that they “threaten the very existence of the company.”

Time

The time it takes for your engineering team to write and maintain custom code will take them away from core business projects that could be driving your business forward.

With an externalized authorization solution, it is possible to reuse access policies, meaning coding it once then applying that code over to the entire application ecosystem. This will allow your team to focus on key initiatives while eliminating the need to write custom code for every complex access requirement.

Authorization vendors have also thought of the pitfalls of implementing the authorization product. That way customers don't have to make mistakes others would when implementing as it has already been worked out.

Resources

It also takes a lot of resources to build your own policy decision engine and write the policies for it. Development teams often don’t have enough internal resources to secure every application, so they are picking and choosing customer-facing apps, while not securing other – possibly back-end – systems.

When an externalized authorization solution is successfully implemented it can set you up to proactively secure applications while your team is building them, instead of making security an afterthought.

Problems are already solved

Security solution vendors who are dedicated to providing authorization already have common problems solved in their solutions. When building in-house the team may not have the resources or expertise to know or think about these things. That means when using a vendor solution you will have to deal with fewer bugs and issues than if you build the software on your own.

A good question to ask yourself is what's your priority - to become a security company or to be a secure company?

Compliance

Solution vendors, in general, are well-versed in compliance rules, laws, and other regulatory requirements as a part of their development process. This could include regulations like General Data Protection Regulation (GDPR), Payment Card Industry Data Security Standard (PCI DSS), and California Consumer Privacy Act (CCPA) which many businesses have to adhere to.

With an authorization vendor like Axiomatics, it is easy to craft and customize specific policies to clearly articulate what conditions your organization requires for users to perform their duties in a manner that satisfies compliance auditors.

This is because they have developed, built, and refined their offerings to incorporate the functionality needed to address both the business/regulation side and from the technical side. In many instances, existing software could be able to meet a customer's compliance requirements, regardless of industry.

Expertise

Building it yourself may be able to create authorization policies, but it lacks authorization expertise. As a vendor, we can guide you through the process of building your authorization policies the right way to protect what matters most.

Maintenance

It all comes back to cost and time. If your team is responsible for maintaining the custom solution, meaning that if it needs to change or the outside environment evolves, it will take several development cycles to get back to normal.

It is also common to see companies abandon their in-house security system when the development team has moved on to other projects. This leaves you with no ability to make changes required by business trends, regulations, or 3rd party integrations. The last point is a notable consideration given that third party external systems like CRM, ERP, payment gateways, etc. are constantly evolving.

Zero Trust

At the core of Zero Trust is the mantra of "Never Trust, Always Verify." This approach aims to increase the security effectiveness and also result in reduced complexity and lower operational overhead.

Taking a step toward successfully implementing Zero Trust is another benefit of an externalized authorization solution as it involves four key capabilities that minimize and simplify common Zero Trust challenges: policy translation, application/database integration, policy modeling, and policy visualization.

A Zero Trust framework can be extremely hard to build in-house which is why even governments reach out for help around Zero Trust. Achieving Zero Trust is a strategy. It's not one technology, it's multiple components that come together to deliver on that strategy.

Authorization is at the heart of a successful Zero Trust strategy as it enables dynamic, policy-driven access decisions. It takes into account attributes from as many data sources as possible to make sure the individual or the machine trying to access a particular resource is, in fact, allowed to do so at this point in time.

As it is dynamic, it considers context to decide if the user can still access a resource or application as the context around the user may have changed in that time period.
As it is dynamic, it considers context to decide if the user can still access a resource or application as the context around the user may have changed in that time period.

Key Takeaways

- Building your own solution will take your engineering team away from core business projects that could be driving your business forward to write and maintain custom code for authorization.
- Maintaining a custom solution takes time and money away from other projects.
- Solution vendors are well-versed in compliance rules, laws, and other regulatory requirements as a part of their development process and can easily incorporate them.
- Authorization is at the heart of a successful Zero Trust strategy as it requires policies that are dynamic.

Deployment Methodology

There are many challenges when introducing an authorization solution. We often see organizations try to implement an authorization strategy without knowing the mechanics of "how." Poor user alignment and engagement generates friction between stakeholders and increases the risk of getting stuck with limited usage adoption and siloed approach to data security.

This is why it is critical to choose a vendor that has the experience and expertise that will help you through common deployment challenges and ensure your success.

Of equal importance, you want a vendor that has fine-tuned their methodology through successful engagements and can discuss the real-world challenges and successes they've seen with organizations similar to yours. This offers you a consistent approach to guide customers through their authorization maturity and adoption.

Create a blueprint for success

No matter the size of your business, a critical path to a successful deployment is a well-tested and repeatable approach to help you successfully deploy your solution. Set a timeframe with your vendor. For instance, understand what a deployment will look like in three months, then six months, etc.

Plan first. Move quickly.

The key to a smooth transition lies in fast and thorough onboarding of all stakeholders - business, developers, security and compliance teams - and alignment on common goals. Communication is key and planning sessions help users ensure the project's internal readiness.

Confidence in your deployment

Ideally, your authorization vendor should leverage what they've learned and design your wireframe as you move into full deployment. At that point, policies are created to ensure authorization policy and data access reports are met, physical architecture is laid out to fit requirements and connectors are built for prioritized policy enforcement points. During this stage, users will start to configure and test policies into existing workflows to measure the impact on the business workflow and overall adoption.

Methodical and flexible

Although the approach is methodical, ideally it will evolve as the vendor you work with understands the business' needs and organizational framework. This means each process should be customized depending on a variety of factors including company size, stakeholder involvement and approval process

The key to success is to educate the entire organization about the authorization platform and its benefits.

Experience speaks

Once fully deployed, both users and IT admins continue to adapt processes to the new system, encourage feedback and report on key success factors and security policies.

More so than other elements of an IAM deployment, authorization requires a strong partnership between you and your vendor. You need a partner that Axiomatics' supports your teams throughout this process and develops a better understanding of your future needs.

Key Takeaways

- The key to a smooth deployment lies in the onboarding of all stakeholders.
- Each process should be customized to the individual organization depending on a variety of factors including company size, stakeholder involvement and approval process.
- The key to success is to educate the entire organization on authorization and its benefits.

Implementing authorization in three phases

To successfully execute an authorization strategy that can seemingly support every application, enterprises must be methodical and thoughtful in the approach.

Growth models are phased approaches to managing change. In the authorization market, this involves solidifying key milestones at each stage of the implementation process. For our authorization strategy, it tends to lend itself well to the crawl, walk, run approach.

Crawl: Aligning to the why

The crawl stage focuses on creating visibility for the value of authorization and how it will support a broader cybersecurity and/or identity vision. This starts with targeting new applications that are in the process of development, instead of trying to transition policies from legacy applications. The recommendation is to target one or two enterprise applications (typically cloud-native) that contain crown jewel information.

Integrations could be complete across the front-end web UI through to microservices, all the way down to the data layer, as needed. This enables the program to target authorization policy deployment as part of their continuous integration/continuous deployment (CI/CD) pipeline.

One of the main objectives of the crawl phase is to create baseline policies and standards that can be tested and refined with these applications. This ultimately lays the foundation for future authorization policies while enabling potential areas of concern to be managed in advance.

This phase also involves gaining momentum and generating anticipation for broader adoption through the measured introduction of the proven business value while conducting training as well as internal marketing.

Ahead of the next stage, the baseline policies and standards should be established with early barriers to adoption addressed. In addition, the impact on securing these applications and data should be proven when put into practice for the target applications.

Walk: securing adoption

The walk stage is centered on securing the broader adoption of the authorization strategy for additional applications. In this phase, we may see instances of transitioning applications with existing authorization policies into this new strategy. Since it is about scaling within enterprise organizations, these applications can be either cloud-based or legacy platforms that contain more sensitive info and require timely attention. Ideally, the walk phase targets application owners or business teams that have a growth mindset and are eager to adopt a more secure way to access critical information.

In this phase, more advanced, risk-based attributes are introduced within the policy based on the framework set in place to define authorization outcomes. The application owners from the previous phase play a significant role as champions, becoming internal references who support the identity teams on broader training and internal marketing initiatives for other application owners by demonstrating how it improves their applications using the initial implementation results as a case study.

Prior to the next stage, best practices for policy authoring and application onboarding are established for ease of execution across the organization. At this point, there should be multiple application owners who have recognized the return on investment for adopting an authorization strategy, with a clear understanding of how it is operationalized for the business.

Run: authorization by design

The run stage is where the authorization strategy is integrated into all facets of the organization, providing a dynamic landscape for constant evaluation and improvement on how to deliver against the strategy. In this stage, the strategy is fine-tuned to continue to address new authorization requirements that may be internal, or externally-driven by compliance regulations.

It will also account for the development of ongoing policies for all layers of the application stack including front-end web, data, APIs and data platforms. At this growth stage, application owners have a command of the policy language and can build their own policies based on the standardized framework within ABAC.

Identity teams will continue with strategic oversight and some policy development, they will lean on application owners for policy definition and authoring.

It is important to note that the run phase is not the final step. It indicates the organization has adopted an authorization-by-design strategy that is constantly evolving based on a mission to protect its critical information.

Considerations for each stage of growth

Every project management toolkit or guide has its best practices to successfully deliver on any initiative. Below are seven elements we recommend you consider defining within the context of the organization's authorization strategy for each stage of growth.

1. Outcome
2. Target scope
3. Resource type
4. Target environment
5. Policy authoring strategy
6. Integration
7. Exit criteria

Key Takeaways

- The crawl, walk, run approach leads to a successful implementation of an Orchestrated Authorization strategy.
- The goal of the crawl stage is to demonstrate the value of the Orchestrated Authorization strategy and establish baseline policies and standards.
- The goal of the walk stage is to widen the adoption of the Orchestrated Authorization strategy by educating the organization and determining best practices for policy authoring.
- The goal of the run stage is to ensure decentralized policy authoring is normalized and all resources can easily be onboarded based on the Orchestrated Authorization strategy.
- The seven key elements of each stage in a maturity model that outline how an authorization initiative is implemented are the outcome, the target scope, the resource type, the target environment, the policy authoring strategy, integration considerations, and exit criteria.

APIs and Microservices NEW

Today, more and more sensitive transactions and data are exposed through APIs. APIs have long been the de facto interface to allow customers, employees, business partners, and services to connect to internal processes, services, and data. In the past five plus years, there has also been a strong incentive to gather, analyze, and monetize data.

While microservices and APIs are the building blocks of modern applications, they rely on developers to follow the best practices for security. Poor developer practices can lead to misuse of scopes, a failure to the principle of least privilege (PoLP) and API sprawl. This is evidenced in the OWASP API Top 10. In particular, the number one security risk as of 2023 is *API1:2023 - Broken Object Level Authorization.*

APIs NEW

In many cases, organizations don't fully understand what is going on with their APIs or even how many APIs they might have, which can open the organization up to unnecessary risk. This is generally known as API sprawl. This risk has caused a shift for enterprises to move away from the usage of subpar API management and access control approaches to mature API access control strategies like fine-grained authorization.

Fine-grained authorization allows enterprises to adopt a comprehensive approach to enforce access control. This means access control policies can be applied to individual assets such as bank accounts (personal, shared, or delegated), patient journals with or without patient consent, or insurance claims that only assigned or trained staff can work on. The same access control policies can be applied at the web application layer, the API layer, and deeper down in the application stack to achieve consistent authorization throughout.

When this is combined with security capabilities such as OAuth and OIDC, organizations can separate the concerns of API protection, authentication, authorization, and delegation, to achieve the fine-grained access required in more complex use cases. Thereby, adding an additional layer of protection that extends beyond OAuth to provide more relevant authorization. It's important to keep in mind that OAuth generally focuses on access delegation, OpenID Connect on authentication, but neither truly addresses fine-grained authorization. Consider the difference between:

- "I, Alice, grant Mint, the finance tracking app, access to my bank account at BMO"; and
- A customer or guardian of the owner can view an account they own.

The former is a typical access delegation (OAuth) use case while the latter is more a policy-driven use case that relies on complex relationships between entities and/or contextual data that OAuth cannot express via claims. This is where ABAC shines.

With ABAC, all access policies are externalized and decoupled from the business layer (application, API, data). This is possible as authorization can be called from the application, API layer, or the API gateway and it allows for updates to access control to be done independently.

This enables organizations to incorporate additional inputs like risk score, device information, location, and more to make an authorization decision before sending the results back to the API for enforcement. It also lets you get a clean audit trail that logs all of the accesses that have been granted or denied within the application.

Additionally, the policies can be authored and managed centrally or by individual teams independently of the application's lifecycle. This gives greater flexibility and visibility into the policy authoring and access review process.

Additionally, the policies can be authored and managed centrally or by individual teams independently of the application's lifecycle. This gives greater flexibility and visibility into the policy authoring and access review process.

When leveraging ABAC, there can be multiple business services protected by a single API, and the policy applied could be dynamic and apply to many/all of them. Combining these solutions will enable the organization to implement an end-to-end API security model that is capable of protecting the privacy of customers and employees.

An added benefit is that authorization can work on the way in and out of the API (or API gateway) to further protect the data flowing out of the services. This is unique to policy-driven authorization: it can only be done with an externalized authorization framework that follows the PEP/PDP principle for runtime/realtime authorization. This opens up the ability to do data redaction, masking, and filtering, all of which are not possible with OAuth.

In turn, this architecture serves the goals of modern security initiatives including Zero Trust by continuously enforcing policies against critical attributes such as risk, time, location, classification, role, etc.

OAuth NEW

OAuth is an open-standard framework that describes how unrelated servers and services can safely allow authenticated access to their assets without the user having to share their login credentials.

It decouples authentication from authorization and allows applications and processes to authenticate and either act on behalf of the user or access parts of the user's data without having direct access to their main authentication credentials. For this reason, it plays an important role in API-centric access and authentication.

However, the specification is meant to formalize the process flow to secure the delegated authentication and authorization process when a consumer interacts with the service provider on behalf of the end-user.

Importantly, the specification does not cover how the decision of whether a consumer's access request should or shouldn't be permitted. This decision is based on finer-grained policies that capture authorization requirements arising from regulatory or business needs. However, the OAuth framework is coarse-grained and does not go into the fine-grained details.

This is because the service provider's inner workings are out of scope for OAuth. Therefore, the service provider must implement and enforce its own authorization policies (or logic) that get triggered when anyone interacts with it.

The tokens presented by the OAuth customer can be considered as attribute values that will be used by the service provider to make an authorization decision based on the defined ABAC policies. ABAC enables the service provider to both model and enforce such authorization requirements that are at a finer grain.

Microservices

The microservice architecture was developed to solve the challenges of a monolithic architecture, which was designed to be developed as a single unit. When designed as a single unit, the monolithic applications lead to issues such as scalability, availability, and maintenance.

To address these problems, a microservice architecture was designed where software is composed of small independent services. This approach can be scalable, consistent, and maintainable. It also makes it easier for applications to scale faster, which enables innovation and accelerates time-to-market for new features.

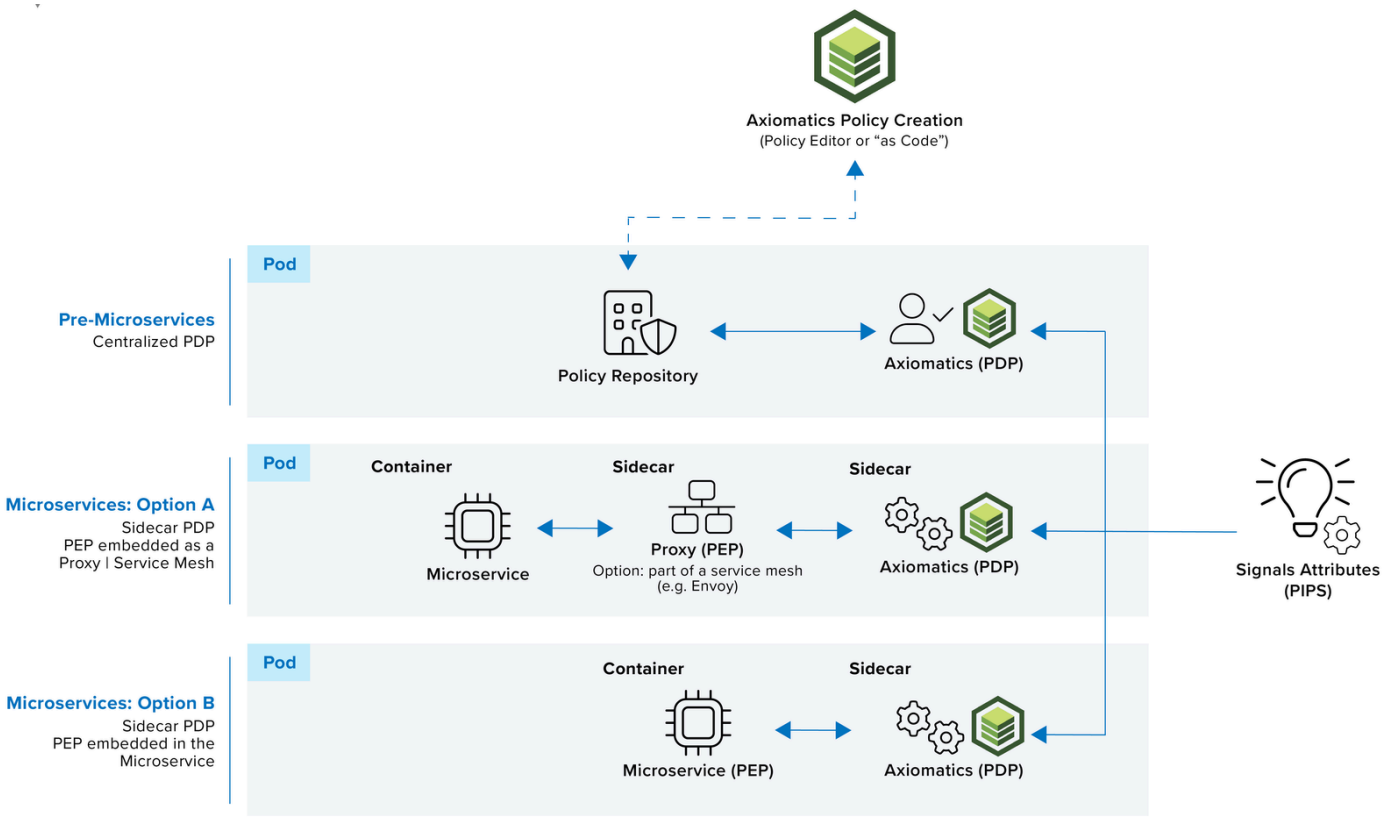
As microservices change the scale of things: you go from having a dozen APIs to hundreds of microservices that require orchestration. When utilizing an authorization solution it is easy to reuse policies across multiple microservices to enforce both business authorization policies and technical policies from an infrastructure side.

While a microservices architecture has varying degrees of component granularity, the goal of dividing the application into individual services means authorization integration must adapt accordingly.

PDPs use policies and attribute data (PIP) to make decisions about whether an attempted resource access should be permitted or denied. The PDP can be provided as a REST/JSON-based microservice built with cloud-native principles. The enforcement can be done either through a proxy as part of a service mesh (e.g. Envoy) or an embedded agent (PEP) in the microservices.

In a traditional monolithic architecture, runtime authorization policies can be served and orchestrated from one PDP. However, as the application is broken down into a microservice architecture, there are multiple options to integrate the PDP as part of the microservice.

Additionally, you could deploy a proxy as a sidecar or as a more centralized service at the node or the cluster level, depending on your scalability needs.



Key Takeaways

- To enforce resource protection over an API channel, the API gateway solution should be complemented and extended with fine-grained authorization.
- Without an ABAC solution, authorization must be hard coded in the gateway – or worse, in the API itself.
- A microservice architecture is scalable and consistent. It also accelerates time-to-market for new features.
- Authorization allows for architecture and deployment compatibility with your microservice-based applications.
- Whether using an API gateway or microservices as an enforcement point, authorization can protect the resources most valuable.

The cloud and authorization NEW

The cloud helps enterprises accelerate application development, as legacy monolithic applications are decomposed to microservice architectures. With that comes a focus on security to ensure the new services are certified against emerging cloud-based security standards. Within that focus, the authorization strategy needs to be reimagined for modern application development.

By being an externalized architecture, policy-driven authorization enables application teams to deploy an authorization strategy where policy decisions scale across a large variety of services. As software-as-a-service (SaaS) and cloud-based applications become predominant in cloud infrastructure, an ABAC model ensures that access control policies are centrally enforced with the creation and governance as decentralized as needed for efficient policy management.

Another benefit of having policy-driven authorization in the cloud is that organizations can take advantage of having a central location where all cloud access control decisions take place. By integrating externalized authorization with the existing cloud & network monitoring services (e.g. Amazon's Lambda, S3, API Gateway, CloudWatch), enterprises can collect and track metrics on virtually every access control request and decision made in the cloud. Therefore, triggers and alerts can then be implemented, to notify someone if too many access request denials are being issued within a given period (as one example).

During the transition to a cloud environment, there is a mix of on-premise applications and cloud services, and while applications are moving to the cloud, a policy-driven authorization solution can apply policies and enforce decisions within both environments, while only managing policies in one location. This hybrid implementation eases the transition of applications into the cloud as the authorization service already exists in the cloud. This saves time, eliminates repetitive re-coding, and shifts focus to the bigger picture of the cloud transition.

Better collaboration through policy-driven authorization NEW

Enterprises invest in collaboration platforms such as Microsoft 365 to empower their workforce to do their jobs better and faster. However, this is not possible with a restrictive approach that offers “all or nothing” security measures that either lock down too aggressively, creating friction and hindering collaboration, or do not offer enough protection, making your organization vulnerable to risk.

Additionally, misaligned access control policies inhibit what employees can share, particularly with anyone outside the organization, resulting in users doing an end run around the system as they download the information and share it outside of corporate control.

To reduce the risk and enable collaboration, enterprises can leverage a policy-driven approach to authorization and access control, creating a secure, frictionless environment for their workforce. This approach means dynamically assessing various attributes in real time, at runtime, to determine permissible file sharing actions, enhancing both security and compliance without being anchored to the application’s update cycle.

Some key features of this include:

Custom policy workflows

Tailor collaboration policies to fit the unique needs of enterprises while enabling efficient job performance and adhering to legal and compliance standards.

User-friendly experience

Offer end-users the ease of intuitive controls within organizational branding, streamlining collaboration and outpacing unsanctioned workarounds.

Data duplication elimination

Centralize data within a secure tenant, leveraging existing security investments to share controlled access that can be subsequently revoked.

The shift towards ABAC and an external authorization architecture empowers organizations with dynamic, fine-grained access controls. This ensures files are securely managed, shared, and protected at the file level while being flexible, allowing for seamless updates and integration with multiple attribute sources. It also offers end users the ease of intuitive controls within the organization’s appetite for risk while streamlining collaboration and outpacing unsanctioned workarounds.

Optimizing current investments

Enterprises are always looking for a way to capitalize on their current investments. When adding an authorization solution to your enterprise it is easy to add a great deal of value to existing investments that your enterprise has made.

Identity Governance and Administration (IGA)

IGA solutions have drastically helped to control and govern access, but there are gaps and challenges that present themselves when trying to scale this approach. The first is that the roles and conditions for which access can be permitted are rarely straightforward in the real world.

Plus, creating policies that have conditions for approving access is quite limiting. Access can be granted if the user is in a certain role, and usually with one or two other conditions at most, such as the device they are using to access, or maybe the time of day.

While the intent to govern and administer appropriate access is in an IGA, there are so many unique conditions that can't be accounted for. The result is that it is often not fully possible to take advantage of IGA solutions in the way they were intended.

Authorization solutions can add a great deal of value to existing IGA investments. By building onto the existing RBAC approach of an IGA, authorization solutions can add an ABAC approach to address a number of scenarios and conditions.

ABAC effectively adds the ability to include a real-time decision tree in the access with RBAC as one of the first decisions in the tree. With this approach, the work and investments already made with IGA solutions and the RBAC approach are not wasted. ABAC adds additional attributes to make a more appropriate decision in real-time.

Recertification

Recertification, also known as access review or attestation, is primarily based on a RBAC approach and is commonly tied into an IGA solution.

Traditionally, recertification is a static approach and only one moment in time – when the process is done. This means if any roles are changed within even a few hours of when the process is completed, this change isn't considered until the next time the recertification process happens. This can leave the organization open to risk outside of when this process is completed.

However, relying on the recertification process in its current form no longer satisfies the intended purpose and moving to an ABAC approach makes more sense for the dynamic IGA needs of enterprises. As an ABAC solution means access decisions are made based on the policies, conditions and context of the moment in time of each request.

Identity and Access management

Over the last decade, thanks in part to multiple compliance mandates and high-profile breaches, organizations have shifted focus and leaned on IAM solutions to ensure workers right-sized access, or only the access they need to only the data they need to do their jobs, and nothing more.

The world is constantly changing and most IAM solutions don't account for additional context associated with other changes. Context is key because without it, IAM solutions can miss key elements necessary to make an appropriate decision about whether to grant or deny access.

An example of this would be if a bad actor tricks an employee into somehow exposing their credentials, as was the case in the 2023 breaches of both MGM and Uber. In a traditional IAM solution that uses authentication and a RBAC policy review, this alone won't necessarily identify any risk of exposure if this has happened. However, when using ABAC as a part of your IAM solution, it will be able to catch the bad actors in real-time and enforce secure authorization dynamically.

Key Takeaways

- Authorization solutions can add a great deal of value to existing IGA investments. By building onto the existing RBAC approach of an IGA, authorization solutions can add an ABAC approach to address a number of scenarios and conditions.
- Recertification is primarily based on a RBAC model and is a static approach. This means if any roles change even within a few hours of when the process is completed, the change isn't considered until the next time the process happens.
- In a traditional IAM solution that uses authentication and a RBAC policy review, this alone won't necessarily identify any risk of exposure if an employee exposed their credentials.
- ABAC when used as a part of your IAM solution will be able to catch bad actors in real-time and enforce secure authorization dynamically.

Generate policy with generative artificial intelligence (AI)

One of the challenges enterprises face is that while/although they understand what authorization is, the value it can bring to the organization by improving security posture and overall secure their business, they can lack certain expertises to get there.

It is the ramp-up time of this domain knowledge that has created a perception of authorization being hard. While many have experience creating policies solely based on roles, they face a challenge when attempting to draft policies with multiple outcomes based on multiple attributes. Axiomatics can help with that, and with the new era of generative AI, there is an opportunity to accelerate learning and the creation of policies while complying with authorization standards and best practices. Generative AI takes authorization from a conceptual strategy to actual implementation, helping to break down the traditional barriers that many have found challenging.

Yes, generative AI is still early and quickly evolving, but we are already seeing opportunities to help address challenges that the industry has been facing since the beginning.

Axiomatic believes that AI when applied to authorization can drastically simplify the requirements gathering and policy authoring as generative AI provides a base level for policy writing. While not a legal source of truth for policy, it can help someone who has never written a line of ALFA or OPA before get an understanding of what the policy needs to include.

This way anyone, without being a developer, can provide initial syntax to the development team. This can help accelerate the development of policies within an organization.

One of the challenges that enterprises face with authorization is that often the business speaks about policies in a more natural/unstructured language whereas developers speak about policies in a more structured/code language. Translating between these two mediums can often be a difficult task. This is where generative AI can help reduce any intimidation that people may feel if they have never written an authorization policy before (either unstructured or as code).

There are potentially dangers here if the syntax provided isn't solid, so vendors have to provide tools to test the syntax to ensure it is correct. This testing process provides a bridge to close what can be a big gap between a developer and business language because sometimes the business may conceive of a policy that sounds good on paper, but won't work in practice.

In the past when someone has said that authorization is hard, it is really the policy development that is hard. However, when generative AI creates this baseline, it is easier to confidently and accurately create policies that can accelerate authorization deployment.

Five ways generative AI will make policy-driven authorization easier

NEW

The market is constantly changing and over the next twelve to twenty-four months various projects will become products as it continues to evolve for today and tomorrow. However, as more information is released on generative AI there are some specific points of how we see generative AI being able to help make policy-driven authorization easier for enterprises.

1 Accelerate authorization time to value

Generative AI accelerates policy authoring as it helps reduce the learning curve for teams that inherit the responsibility of deploying software to avoid the “deployment chasm”. This means that a process that normally takes days and weeks to learn is now reduced to minutes with generative AI.

2 Empower anyone to write modern authorization requirements

Defining authorization policies based on ABAC can be confusing, but generative AI can be a great coach when it comes to defining policies. This helps ensure that policies are well defined and contain attributes and conditions as opposed to being vague in what the goal of the policy is to do.

3 Convert natural language policy to/from machine readable code

Application owners and other business people speak a different language compared to developers and architects. Generative AI can help translate natural policy language to machine readable code. Now policies finally make sense to both parties at the same time.

4 Enhance authorization policy analytics and access reviews

It enhances policy analytics and access reviews as it can speed up the lengthy process of access reviews and as it runs continually in the background it catches errors that the traditional process doesn’t see.

5 Provide translation between authorization languages

Authorization languages are use case driven, so in some cases, there can be multiple languages used within one organization. It can be difficult to translate between languages, but generative AI can be used to translate between the different languages.

Generative AI and authorization vendors

When working with a vendor who claims to be using AI, make sure that vendor is taking a pragmatic approach and is not promising a fix-it-all solution. At this point in time the technology is still early in its adoption. Certainly at an enterprise level, there is no one vendor that can say theirs is the only solution to have this figured out. The other considerations around this are privacy and security - everything around AI is still being figured out right now. When looking for a vendor that is using AI, enterprises want to look for a vendor that has a phased approach and a clear long-term product roadmap for incremental advances that leverage the technology.

Ease of use for the business user is not often a consideration for vendors who develop authorization policy solutions. Instead, almost all solutions require a level of coding experience that means the uptake with anyone of a business focus is really quite limited (or non-existent). However, we feel that the job of authorization vendors should be to provide all users with the tools that they need based on the experience they have, and generative AI can help to bridge that gap.

But generative AI will take away some of those big barriers that enterprises have when looking at an authorization project. It enables them to test the waters and, in many cases, get rid of the “it’s too hard” arguments.

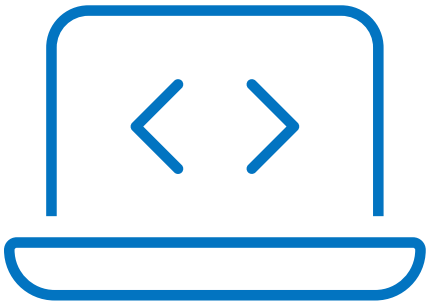
Key Takeaways

- Generative AI is still in the early stage, but we are already seeing opportunities to help address challenges that the industry has been facing since the beginning.
- Generative AI takes authorization from a conceptual strategy to actual implementation, helping to break down the traditional barriers.
- AI when applied to authorization can simplify the requirements gathering and policy authoring as Generative AI provides a base level for policy writing.
- Generative AI creates a baseline, so it is easier to confidently and accurately create policies that can accelerate authorization deployment.
- When working with a vendor who claims to be using AI, make sure that vendor is taking a pragmatic approach and is not promising a fix-it-all solution.

AXIOMATICS

Take the next steps

Meet with our solution experts for a demo of our Orchestrated Authorization solution and discuss how our approach can save your organization time and money.



Stay connected and informed

Get the latest insights, announcements, and assets from our thought leaders and customer success team:

- Blog articles
- Press releases and announcements
- Resources, solution briefs, and tools
- Join us on LinkedIn
- Subscribe to our Linkedin Newsletter
- Subscribe to our YouTube channel

Axiomatics is the originator and leading provider of runtime, fine-grained authorization delivered with attribute-based access control (ABAC) for applications, data, APIs, and microservices.

The company's Orchestrated Authorization strategy enables enterprises to effectively and efficiently connect Axiomatics' award-winning authorization platform to critical security implementations, such as Zero Trust or identity-first security. The world's largest enterprises and government agencies continually depend on Axiomatics' award-winning authorization platform to share sensitive, valuable, and regulated digital assets – but only to authorized users and in the right context

Follow us on LinkedIn: https://www.linkedin.com/company/axiomatics

Join us on YouTube: https://www.youtube.com/@axiomatics

Learn more or request a demo of our solution:

axiomatics.com

info@axiomatics.com

US Office: +1 (312) 374-3443 | Europe Office: +46 8 51 510 240

AXIOMATICS