



WHITE PAPER

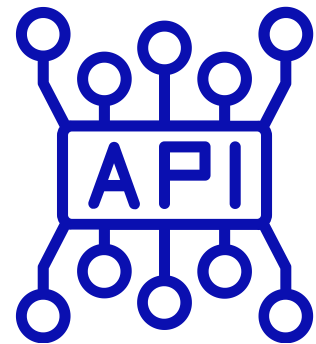
Protecting microservices and APIs with ABAC, OAuth, and OpenID Connect

[Attribute Based Access Control \(ABAC\)](#), OAuth 2.0, and OpenID Connect (OIDC) are complementary standards that can be used individually or in concert to offer comprehensive access control for applications that are built using microservice and API approaches. This document outlines a set of examples where security standards work together in addressing requirements that are difficult or impossible to meet by using OAuth or OIDC alone. We assume that readers have knowledge of ABAC, OAuth and OIDC standards and therefore do not include basic information in this document.

Overview of API protection

Today, more and more sensitive transactions and data are exposed through APIs. A primary driver for this movement is the digital transformation and innovation initiatives that many organizations undertake. APIs have become the de facto interface to allow customers, employees, business partners, and services to connect to internal processes, services and data.

While [microservices](#) and APIs are the building blocks of modern applications, they are still subject to sins of the past by developers if they don't follow best practices for security. Poor developer practices can lead to misuse of scopes (explosion in number of scopes required), not adhering to principle of least privilege (accidentally or maliciously exposing too much data) and API bloat (including security code in the application instead of calling a security service). Such practices can lead to longer development cycles, more costly application maintenance, complex security administration and lack of visibility for audit/compliance.



[API gateways](#) are often an essential component of an API protection architecture. Most API gateways provide internal security capabilities or integrate with an external Identity Provider (IdP) that handle authentication, delegated access and other requirements. These security solutions support various standardized access control mechanism such as OAuth, OIDC and SAML to validate identities, manage/validate tokens and support other scenarios. The basic security capabilities of the gateway may be sufficient for simple or basic use cases, but organizations that safely need to expose and share highly-sensitive data likely will need additional capabilities.

In order to enforce resource protection of who can perform certain transactions or who can access and use specific data over an API channel, the API gateway solution should be complemented and extended with a fine-grained authorization solution. By extending the API Gateway with a dynamic, [policy-based authorization solution](#), organizations will be able to enforce resource specific access control. This means that the access control, for example, can be applied to individual documents, bank accounts (personal, shared or delegated), patient journals with or without patient consent or insurance claim that only assigned or trained staff can work on.

By combining security capabilities such as OAuth and OIDC with a dynamic authorization solution based on the ABAC standard, an organization can separate the concerns of API protection, authentication, authorization and delegation. This combined solution will enable an organization to implement an end-to-end API Security model that is capable of protecting the privacy of customers and employees, the most business-critical transactions and the most sensitive data across the API channel.

OAuth Scopes are an excellent mechanism to implement delegated access control. But OAuth Scopes are quite static and do not support any language to express authorization policy. Writing authorization policy logic using OAuth and OIDC will result in organizations embedding authorization logic into the APIs. This creates a tight coupling between the API logic with the authorization logic which makes them difficult to manage and audit. OAuth Scopes and OIDC are therefore mostly suited for coarse-grained and functional access control e.g. which users can perform payment transactions or can view patient journals.

Baseline architecture #1: API gateway and OAuth 2.0

In Figure 1, a typical deployment configuration consists of the OAuth 2.0 Authorization Server and an API gateway playing the role of Resource Server - we make the assumption that this function is separated from the Business Service. You can think of this as legs two and three of an `authorization_code` grant flow, where the access token is issued to the client, which is attached on subsequent calls to the business service.

The API Gateway validates the access token before forwarding the request to the business service. OAuth alone doesn't provide a good mechanism for enforcing policy for business services that contain sensitive data, represented by the PII label, a common scenario in industry. In the next section, we will describe how the ABAC service can complement the OAuth flows by providing a comprehensive policy evaluation service.

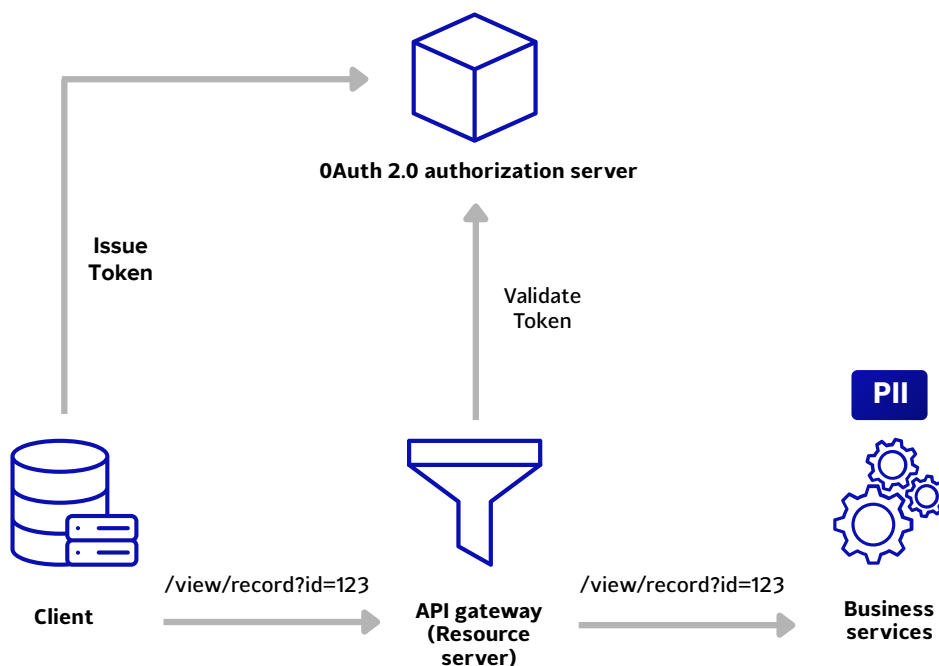


Figure 1: API gateway interacts with OAuth 2.0

Baseline architecture #2: API gateway and ABAC service

An API gateway plays a key role in API/microservice deployments by providing many security, management and operational capabilities. When advanced access control requirements must be met, the gateway can easily be configured to call an ABAC service to determine if access to an API should be granted or denied, as depicted in Figure 2. All access policies are centrally managed, and enforced, in the ABAC service instead of hard coding this logic in the gateway – or worse, in the API itself. There could be multiple business services protected by a single API Gateway and the policy applied could be dynamic and apply to many/all of them.

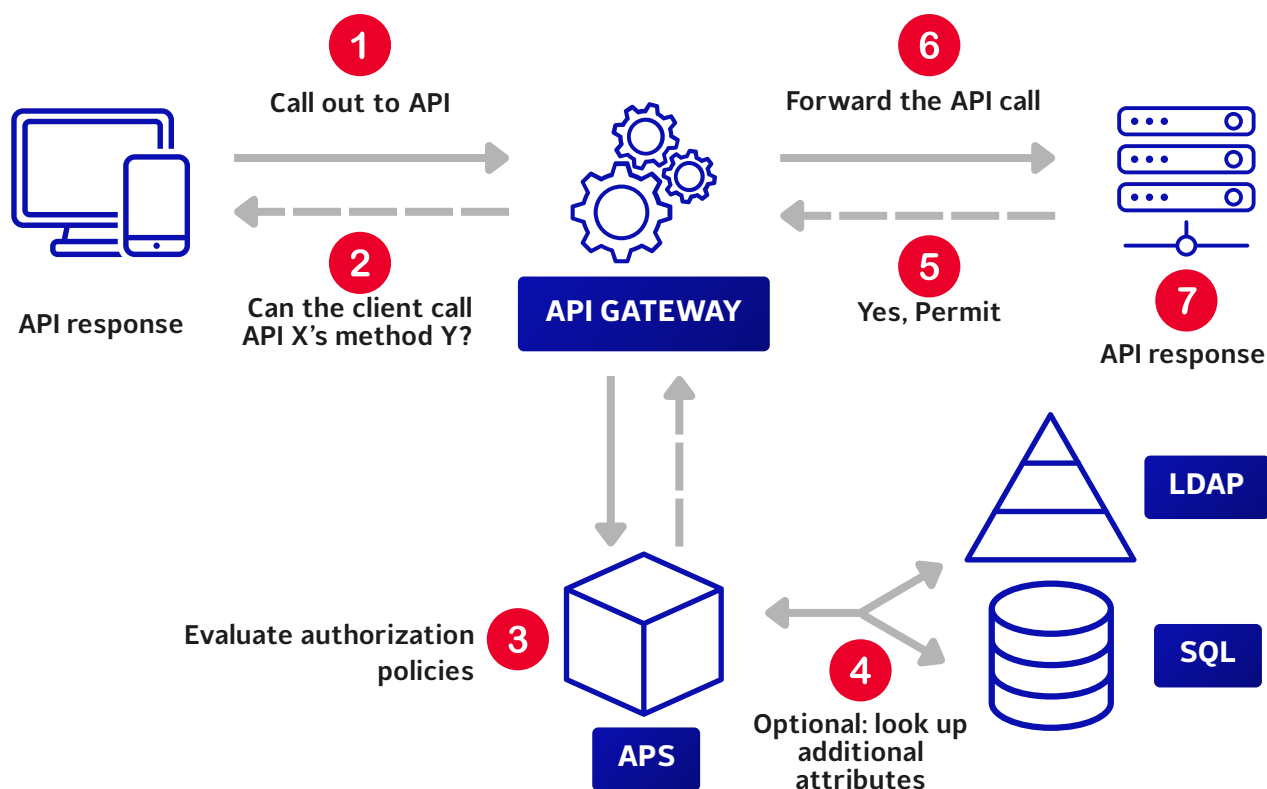


Figure 2: ABAC architecture with API gateway as the enforcement point

In a few simple steps, the API gateway can be configured to call an authorization service for scenarios where additional policy evaluation is needed. For these scenarios, the gateway constructs a formatted JSON message with pertinent data (subjectID, method, authentication strength, requested resource, etc.) and POSTs it to the REST endpoint of the authorization service. During policy evaluation, the authorization service can incorporate additional contextual information such as determining the relationship of subjectID to the requested resource (if any), checking current client status (bronze, silver, gold), weigh the current risk score, and so on. The resulting decision is sent back to the gateway where it is parsed and enforced. No custom coding is required for this level of integration, only configuration of the gateway.

API gateway calls ABAC service using OAuth information

All the ABAC and OAuth/OIDC pieces come together in this configuration option. In Figure 3, the API gateway has a lot of information at its disposal just before performing the API call on behalf of the client. For example, the gateway has scope information and, optionally, has called the OIDC user info endpoint to collect additional data on the logged in user in a JSON Web Token (JWT). The attributes about the user along with the attempted API are now packaged in a message to the ABAC service for evaluation. The JWT token could also be forwarded in raw format to the PDP, thereby reducing configuration of the gateway.

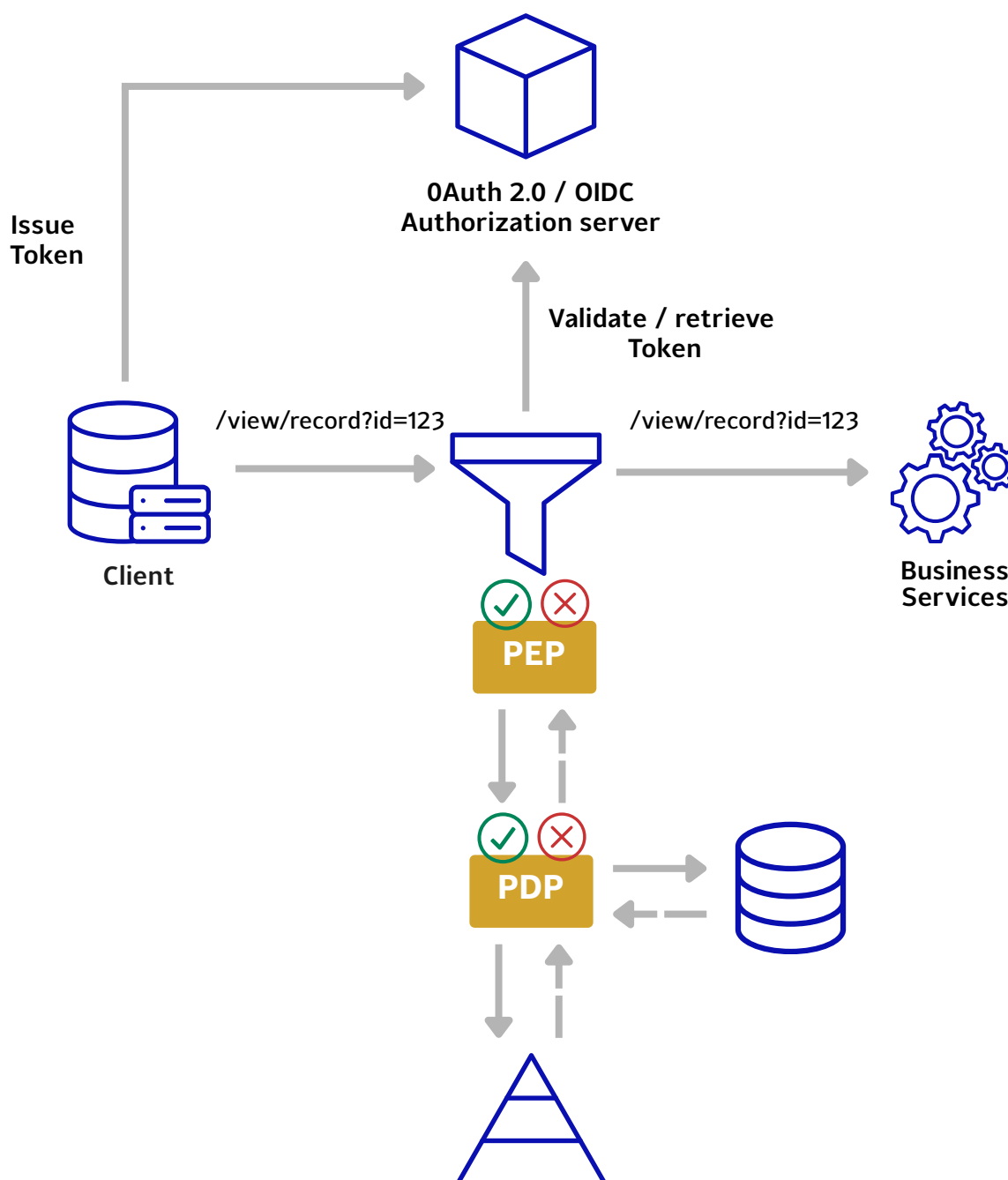


Figure 3: API gateway mediates ABAC – OAuth data flows

As noted earlier, the ABAC service can access any additional context (risk score, device information, etc.) in order to make an authorization decision before sending the result back to the gateway for enforcement.

Filtering messages on the API response leg

API security is not a one-way street! The previous scenarios are all focused on applying authentication and access control on the incoming API call. What about the response message? If the API call is to retrieve a data record on a student, banking, or health record; there may be sensitive or private data elements that should be filtered out depending on the authority of the caller. Therefore, you can configure the API gateway to call the ABAC service for determination of any field filtering that should be applied before returning the record to the calling user or application. In this case, the gateway sends metadata (such as region, PII flag, etc.) from the data record to the ABAC service for evaluation. The ABAC service returns a set of decisions or filter pattern that the gateway applies to the data record.

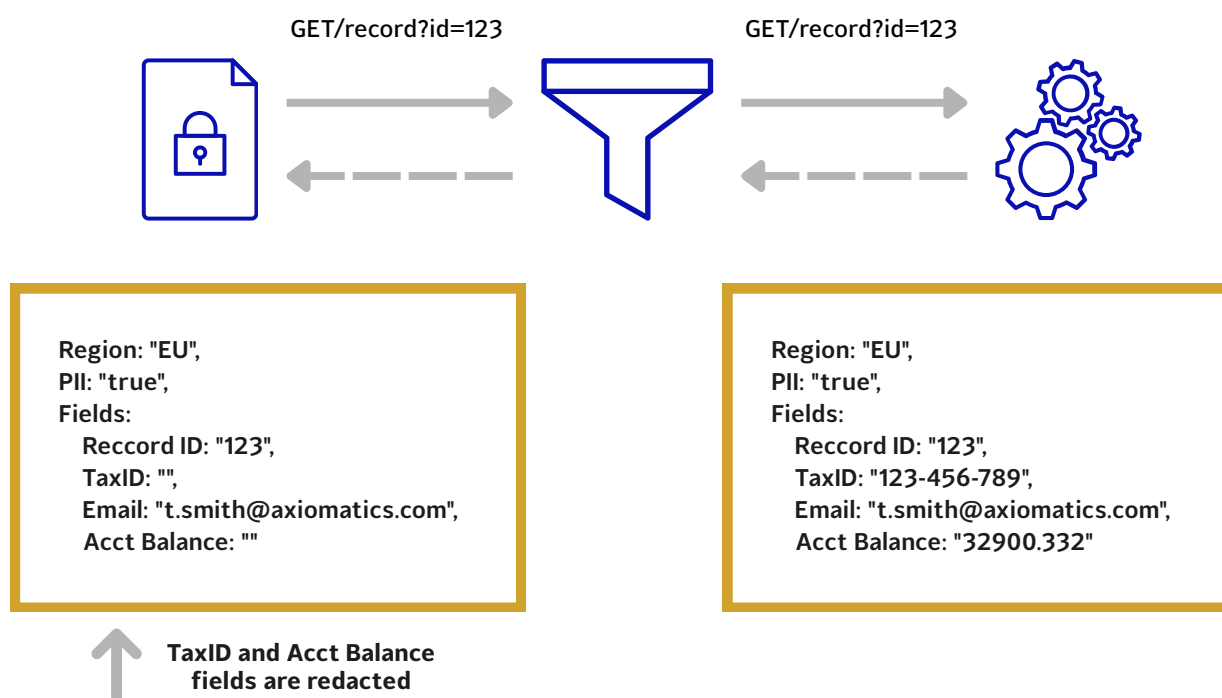


Figure 4: field filtering on the API response message

ABAC as a microservice

In keeping with the API/microservice theme, it is important to consider the ABAC authorization service as a microservice or set of microservices. This point is relevant because it allows for architecture and deployment compatibility with your microservice-based applications.

For example, the ABAC service has the typical characteristics of a microservice:

- **Is stateless and immutable**
- **Has a well-defined interface**
- **Supports REST/JSON**
- **Has a bounded context**
- **Is Fault tolerant**
- **And container friendly**

These characteristics blend well with a [DevSecOps](#) philosophy that supports automated deployments, high availability, and continuous delivery models.

Performance considerations

System performance, response time, and capacity are always a consideration for administrators and operations teams. Sometimes trade-offs must be made between topics like performance, security, risk and user convenience. That said, Axiomatics has many years of experience in making ABAC systems perform extremely well in very demanding customer use cases. We can provide a lot of expert guidance on system configuration, [policy modeling](#), and cache settings.

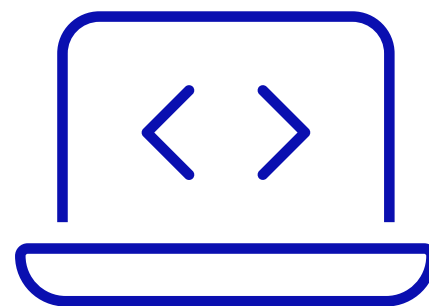
If all of the identity, context and resource information is available to the API Gateway at the time of policy enforcement and the Policy Decision Point (PDP) doesn't have to supplement from another system, the performance cost of externalizing authorization becomes almost negligible.



Take the next steps

Learn how a security vendor can help secure your access control while allowing you to focus on your core business initiatives.

[Meet with our solution experts for a demo](#) of our Axiomatics Authorization Management Platform 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 - a Leonardo Company 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 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://linkedin.com/company/axiomatics>

Join us on YouTube: <https://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