Why **Policy based** is better than Role base in support of a business case?

Identity and access management (IAM) is typically responsible of the identity life cycle management within various systems the user needs to access, including on-boarding, off-boarding and role changes. But, even though IAM solutions are in the market for more than 30 years, they are considered to be extremely complex and highly time and resource consuming.

Adding to that is the ongoing expansion of the organization, both with its data and identities. Many organizations are considering or have already expanded their data to the cloud. What was once controlled and secured by the internal measures of the organization is now distributed and so is the control of its access.

Many organizations are also required to support distributed identities. The organizational systems are accessed by the employees from anywhere, office, home or mobile. Mergers, external contractors, extend even more the sources of identities, those systems are required to support.

We sort out the mess associated with authorizations & access management in fast-paced organizations.

The IAM building blocks

The IAM building blocks can be broken down to the following three:

- Identity – How is the online defined and managed?
- Authentication (AuthN) – How can the identity be proven?
- Authorization (AuthZ) – What can the identity do?

Existing technologies focus primarily on the first two, Identity life cycle management and Authentication. Authorization is usually left to the responsibility of the developer, the application. The result is a half-baked solution, and fully baked AuthZ decisions in the code. No real control, and no visibility of what can the user do or see.

It's time for a change, it's time for a better way to handle AuthZ.

The developments in the AuthZ approach are aiming to simplify the AuthZ process, enable it to scale faster and provide much better control and visibility to the organization of its assets.

We Authorize!

info@plainid.com www.plainid.com

Why **Policy based** is better than Role base in support of a business case?

Out with RBAC in with ABAC

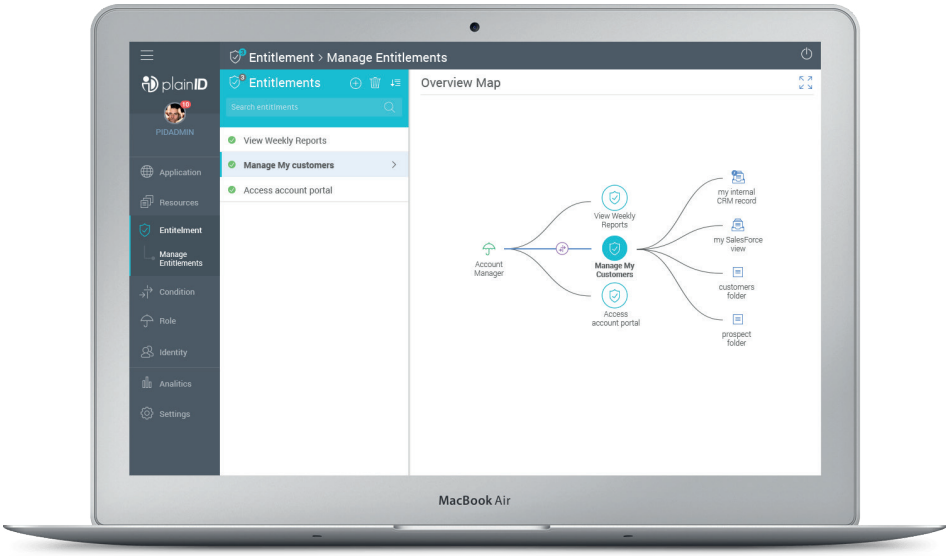
Assigning access controls is one of the foundational steps in information security and compliance. Enterprises commonly deal with data—personal, commercial, business internal —that absolutely cannot leak to the public at large nor internally within the organization. All the major compliance regimes, such as FISMA, PCI-DSS, and HIPAA, have detailed requirements that lay out who can access certain data, when that data can be accessed, and how to keep records of that access. One last question remains, however—how do you ensure that no unauthorized persons can access that data?

In order to truly lock down data, the NIST, via its National Cybersecurity Center of Excellence (NCCoE), is pushing forward a more flexible method to manage privileges - Attribute Based Access Control (ABAC).

Modulate Access Using Attributes

Under **ABAC**, access to a particular record or resource is moderated based on certain traits—attributes—of the person accessing the file (the subject), the resource itself (the object), and the time and place where the object is being accessed (the environment). Attributes of the subject may include their title, certifications, or training. Attributes of the object may include its related project, the personally identifying information (PII) it contains, and the sensitivity of that PII. Viewed holistically, these attributes can now be used to set the rules that would enable access to data and resources.

RABC (Role-Based Access Control) on the other hand, involves creating a role for every organizational or business functionality, giving that role permission to access certain records or resources and assigning a user to the role. This system is entirely too granular, not flexible and very limited in large scales.



info@plainid.com www.plainid.com

We Authorize!

Why **Policy based** is better than Role base in support of a business case?

Simplify Access Controls and Smart up your Permissions

Under ABAC, the number of actions needed to enforce access controls is substantially reduced. Blanket policies can be set, using natural language. For example, “Only auditors can view the sensitive data associated with their assigned projects” is a very simple policy to apply and manage under the ABAC system. Handling large amounts of users and data is relatively easy, since they all automatically fit the policy.

Additionally, environmental conditionals can be used to further modulate access. For instance, maybe a set of files is so sensitive that it can only be viewed on desktops that are on the corporate site. Environmental conditions can be based on a location, time, risk of access, and even events. For example, you might want to provide limited access upon a cyber-security event, and extend access of corporate engineers in case of an approved incident.

Using ABAC, enterprises can easily comply with GRC requirements that requires them to segregate data access, prevent unauthorized users from accessing sensitive data, control access creep, and even prevents authorized users from accessing sensitive data in a risky manner.

Automation to cope with constant changes

Change of role, organizational structure, assignment to new projects – adapts immediately to what documents, medical records, servers and more the user can access. No additional action is required to enable access to the new project data, the new user in the department, or a freshly assigned accountant.

ABAC truly is a more efficient approach to support your access decisions.

Attributes are now "how we role"

By 2020, 70% of all businesses will use attribute-based access control (ABAC) as the dominant mechanism to protect critical assets, up from <5% today.

Gartner

info@plainid.com www.plainid.com

We Authorize!

Why **Policy based** is better than Role base in support of a business case?

How Do You Authorize:

The Old Static Way or the New Dynamic Way?

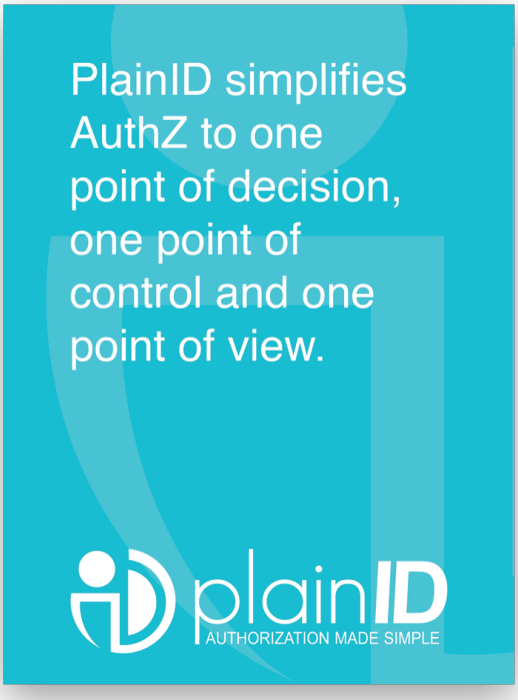
Authorization (AuthZ), is the core of organizational security—which also means that it's central to productivity, AuthZ determines what a digital identity can do within each and every application.

Employees need access to data and applications anytime and while on the go, but for security reasons, they can't have that access at all times. So, when should that decision be made? When is the right time to decide if access is permitted or denied?

Traditional AuthZ – Once upon a time…

It's common for authorization decisions to be triggered by HR movements, for example when on-boarding a new employee, changing an employee's role or terminating employment.

When a new employee joins an organization, they are likely to receive the same AuthZ as a co-worker, gain additional permissions over time, and that would be the end of it. Authorizations aren't changed or even reviewed as more factors change.



Is that enough? Of course it's not enough

The expectations of core security within organizations today are higher, they need to be smarter and to be able to take more factors into account, such as where the employee is accessing from, whether their role has changed, or if the parameters of the project have changed. A definition that was made a year ago, or even an hour ago may no longer be relevant. When gaining access to the most valuable assets of the organization, the decision needs to be made now!

Classic AuthZ methods have limited functionality. They rely on repository-defined groups or roles that provide a link between users and resources. Those authorization decisions are preconfigured and cannot change in real time. Giving a user authorization to view and use a certain suite of files and applications means that unless the administrator manually revokes authorization, the user will be able to use them forever.

We Authorize!

info@plainid.com www.plainid.com

plainID
AUTHORIZATION MADE SIMPLE

Why **Policy based** is better than Role base in support of a business case?

Smarten-up your AuthZ

Dynamic AuthZ represents access decisions that are made in real time. When the user requests access to data, tries to manipulate data, or attempts to use an application, Dynamic AuthZ can consider what factors make those actions a potential risk. For example, the time of access, the location, the worker's employment or vacation status, and the security status of the system are all deterministic factors that delineate a user's ability to take certain actions.

What could this look like? Access to modify research data is permitted for authorized users until that data is approved; after approval, the data can only be viewed. Or, let's say that normally, access to IT resources in a production environment is limited, but, upon an incident that requires specific attention, access is approved for the authorized user. Dynamic AuthZ can be used to enable access, to prevent access based on what happens in real time, who the user is, and where they are accessing from. It lets you make smart decisions on how data and resources will be available to be used.

Dynamic AuthZ: Connect with the 'here and now' to protect your assets

The Unsudden Death of Group-based Access Control

It's true that traditionally, security groups are the most common way used to authorize access to data and actions. But what is truly their role? How do they operate? Security groups, or security roles act as a link, an intermediate between the users and the data, information or actions. They do not directly reflect what a user can actually do or see.

Groups Are Old School

For example: we want developers to access development data, and managers to access management data. So we create one group for developers and another for managers, and assign the users accordingly. But who is responsible for ensuring that the development group can only access development documents? And that the management group can access only management data? What happens if a development manager, who is assigned to both groups, accidentally places an "Employee Management Assessment" document in the development folder?

info@plainid.com www.plainid.com

We Authorize!

plainID
AUTHORIZATION MADE SIMPLE

Why **Policy based** is better than Role base in support of a business case?

Diverted Responsibility

When using group based access, the responsibility is diverted. The IAM team usually connects users to groups, but the data and activities this group can access is under the responsibility of the application or the business owner. In practice, users often receive access to too many resources that they do not need and are prevented from receiving access to specific resources and tools that they do need.

This Approach is Not Good Enough

Firstly – it leads to many errors and, secondly – it does not scale.

Reverting to the developer example: Whenever that developer joins a new project, they need to be assigned to a new set of documents, tools, and permissions. In addition, access to their previous set of tools etc., will probably need to be revoked. This may be manageable if there are only a few developers in the organization, but what happens in places with hundreds or thousands of employees? The IAM team will spend a large and unnecessary amount of time unpacking and reassigning permissions even if only a single employee changes roles.

A Better Way: Connecting Users Directly to Data

Resource-based access allows you to connect users to the data they are entitled to access. Without a middle man, without giving away responsibility, without any unknown stages which could lead to potential errors.

For example, in resource-based access, access can be granted based on the matching project identifier for both user and document. It can also be based on the user's role in the project, with access determined according to project stage – Project A is in review stage, so its data is accessible to all reviewers that are assigned to this project.

Resource-based access enables direct connection and accurate visibility to what a user can see and do.

Empty the Repository: Why Virtual Tokens are Better for AuthZ

If you're using a business application, it is very likely to have a user repository attached. This is usually a simple database containing an ID and a list of authorized actions for each user. It's a simple system, that requires IT resources to be managed.

Several solutions have been tried, with LDAP (Lightweight Directory Access Protocol) as the most popular. This is, in effect, is a single directory designed to share user and authorization information between many applications. Its advantages are that it is an industry standard designed so that every developer can freely integrate it into their product. The drawback, however, is that it didn't fit all AuthZ needs and so wasn't widely adopted.

We Authorize!

info@plainid.com www.plainid.com

plainID
AUTHORIZATION MADE SIMPLE

Why **Policy based** is better than Role base in support of a business case?

The Problems with Repositories Mimic those of Static AuthZ

In addition to the problem of volume, repositories have drawbacks common with other traditional forms of AuthZ.

- Administration: In order to change permissions for a given application, the repository needs to be updated. Either manually or by a provisioning system, in both cases it's a complicated task that requires time and resources.
- No Flexibility: Authorizations don't change based on any variables. For example, a cyber security event, or user login through a mobile device, won't remove any assigned permissions. Repositories are static, however, and their users & permissions must be programmed in advance.
- Inefficient Distribution: With over 500 repositories in the average enterprise, the problem isn't just a matter of scale. It is difficult to apply AuthZ policy consistently over such a large volume of databases. If the AuthZ policy isn't applied consistently – whether due to accident or indifference, then certain applications may become security risks.

> *"With "healthcare Y2K" looming and accelerated demand, seven of 10 CIOs are rolling out more enterprise apps during 2013 (46 on average) to keep pace with frontline business needs."*
>
> *http://www2.delphix.com/cio-outlook-2013*

Virtual Tokens Provide the Answer

Virtual tokens take one of the traditional aspects of AuthZ and flips it on its head. What if, instead of storing authZ information in large repositories within each application, you instead reduce it to a small repository fitted for an individual user? This is what a virtual token represents. Upon access, this token is sent to the application, which responds accordingly.

This approach displays some marked advantages over the traditional repository approach. For one, it's responsive -the data carried by the virtual token allows the application to respond dynamically based on conditions described by the AuthZ token. Secondly, virtual tokens are allowed to be small, containing only the information that's necessary for the app to authenticate and authorize the user.

Say NO to Provisioning

Lastly, virtual tokens reduce the need to maintain all those repositories, so no more unmanaged AuthZ, no more "ghost" IDs.

We Authorize!

info@plainid.com www.plainid.com

Why **Policy based** is better than Role base in support of a business case?

What PlainID Does

We sort out the mess associated with authorizations and access management in fast-paced organizations.

PlainID simplifies AuthZ to one point of decision, one point of control and one point of view. Our agile, standards-based platform reduces the many required resources that are needed to drive new projects, new ideas and new revenue. PlainID lets business owners control and fine-tune access by providing a clear view and understanding of every authorization level. Companies that use PlainID benefit from a simplified AuthZ platform and meet the demands of growth without worry.

PlainID quickens new implementations by meshing emerging authorization standards with existing technologies. On premise, in the cloud or mobile based implementations are supported.

It really is mess-less

REQUEST YOUR DEMO

The rapid demands of business require a clear path to the tools and apps that enable growth. We don't put up roadblocks; our solution is not messy or hard to understand.

This truly is a fresh approach to prevent time wasting on Identity and Access Management.

www.plainID.com

We Authorize!