ALTICE LABS WHITEPAPER

Identity and Access Management

December 2014

Keywords: SAML, OpenID, OAuth, XACML, Identity, Authentication, Authorization, Accounting, Federation, Auditing, Meta-Users, Meta-Attributes, Stores, RBAC, Roles, Access

altice labs

Altice Labs
Rua Eng. José Ferreira Pinto Basto
3810-106 Aveiro – Portugal
http://www.alticelabs.com
Tel: +351 234 403 200
Fax: +351 234 424 723

Contents

Executive Overview

Identity Management has evolved over time by externalizing individual security services such as identity stores, provisioning and authentication.

The Altice Labs strategy for identity management delivers a comprehensive solution to manage identities, credentials, and identity-based access policies across heterogeneous environments. Altice Labs Identity and Access Manager product materializes that strategy in a suite of modular components that can be deployed separately depending on the client needs. A client for Altice Labs IAM product may vary from the developer that needs to outsource identity concepts (i.e. authentication, authorization and user management) to a third party, to an enterprise that has a huge complexity and heterogeneity of systems and identity stores. In such scenarios have several replications of the user's information, spread in different stores. These problems may leads to data incoherence and difficulties in terms of single sign-on, user's identity management and auditing trials.

1. Introduction

Identity Management is the combination of business process and technology used to manage data on IT systems and applications about users. Managed data includes user objects, identity attributes, security entitlements and authentication factors [15]. Within the enterprise, an identity management system comprises a system of directories and access control based on policies. It includes the maintenance of the system (adds, changes, deletes) and generally offers authentication, authorization, single sign-on and auditing.

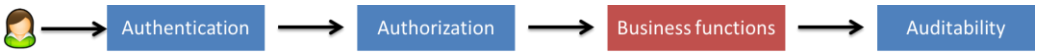

Figure 1 – IAM *generic* functions

Figure 1 shows, at blue, the three main functionalities of an IAM system. The user is authenticated and authorized to access business functions (i.e. services). All the related operations feed the auditing system that may be used to fraud detection or to guarantee that security constraints, standards (i.e. ISO 27000 series), or laws (i.e. Sarbanes Oxley [16]) are been taken into consideration within the enterprise.

The Burton Group defines identity management as follows: "Identity Management is the set of business processes, and a supporting infrastructure for the creation, maintenance, and use of digital identities." [17]

IAM tackles the end-to-end life cycle management of digital identities. An enterprise Identity Management solution should not be built of isolated silos of security technologies, but rather, should consist of integrated technologies. Nonetheless IAM is not just about technology, it comprises three essential elements: **policies**, **processes** and **technologies**. **Policies** refer to the constraints and standards that needs to be followed in order to comply with regulations and business best practices; **processes** describe the sequences of steps that lead to the completion of business tasks or functions; **technologies** are the tools that help accomplish business goals more efficiently and accurately while meeting the constraints and guidelines specified in the policies. This relation is essential to understand the IAM ecosystem is to clearly apply such systems to enterprises. The relationships between those elements may be represented as a triangle (Figure 2), where a feedback loop that links all three elements together.

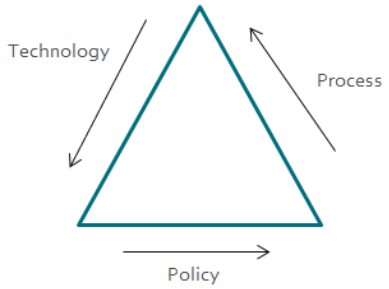

Figure 2 – Essential elements of an Identity and Access Management system

Every organization is different and the right mix of technologies, policies and processes for one company may not necessarily be the right balance for a different company. Therefore, each organization needs to find its own balance represented by the uniqueness of its triangle.

Organization's IAM system does not remain static over time. New technologies will get introduced and adopted; new business models and constraints will change the corporate governance and processes to do things. IAM is a journey, not a destination.

1.1 Digital Identity definition

Personal identifications in today's society can take many different forms. Some examples of these forms are driver licenses, travel passports, employee cardkeys, and club membership cards. These forms of identifications typically contain information that is somewhat unique to the holder, for example, names, address and photos, as well as information about the authorities that issued the cards, for example, an insignia of the local department of motor vehicles [17].

Identities in the physical world is fairly well understood, the same cannot be said about the definition of digital identities. Digital identity consists of the following parts:

- **Identifier** – A piece of information that uniquely identifies the subject of an identity within a given context. The context defines the boundary inside of which an identity is used and it is valid. Examples of identifiers are email addresses, X500 distinguished names and Globally Unique Identifiers (GUIDs).
- **Credentials** – Private or public data that could be used to prove authenticity of an identity attribute (or claim). For example, Alice enters in a password to prove that she is who she says she is. This mechanism works because only the authentication system and Alice should know what the password for Alice is. A private key and the associated X509 public key certificate is another example of credentials.
- **Core Attribute** – Data that help describe the identity. Core attributes may be used across a number of business or application contexts. For example, addresses and phone numbers are common attributes that are used and referenced by different business applications.
- **Context-specific Attributes** – Data that help describe the identity, but which is only referenced and used within specific context where the identity is used. For example, within a company, the employee's preferred health plan information is a context specific attribute that is interesting to the company's health care provider but not necessarily so to the financial services provider. [17].

1.2 Drivers for Identity and Access Management

The use of an IAM product needs a clear understanding of what they intend to achieve and what should not be expected. As said previously, it is a complex process that encloses three essential elements and a several functionalities, where “Single Sign-On” (usually used as a synonymous with IAM) it is only a nice-to-have in the larger set of functionalities and possibilities. IAM encloses much functionalities that just the Single Sign-On functionality.

In a simple way, the drivers for IAM usually spin around three considerations:

- **Risk & Compliance**
  - Secure information assets and restrict their access only to legitimate users through authentication and authorization, and to protect against business, legal and reputation risk arising from inappropriate access.
  - To ensure compliance with enterprise security policy across all applications and information assets (i.e., through password policies, role-based access control, etc.) and meet internal and external audit requirements.

- To ensure accountability through role-based access, approval processes and audit trails of relevant user activity (e.g., logins, failed logins, application accesses, etc.).

- **Cost Reduction**
  - Reduce the effort involved in manual provisioning, de-provisioning and user management, through automation and self-service, especially with increasing volumes.
  - To eliminate or reduce the cost of errors, delays and inefficiencies arising from manual processes and other elements of waste (e.g., orphan accounts, unused storage, etc.)
- **Convenience**
  - Provide a Single Sign-On (SSO) environment to users (eliminating the need to remember multiple sets of authentication credentials).
  - To expedite operations through self-service features (i.e., password reset/forgotten password, delegated administration, etc.). Some organizations also hope to achieve a “Single View of Customer” through implementing IAM, which allows to have a clear understanding on customer behavior better and to enable up-selling and cross-selling of products.

2. State of the Art

2.1 Drivers for Identity and Access Management

Identity and Access Management is an activity that is concerned with the control of access to resources and assets (ICT or physical resources) that is accomplished by allocating and authenticating the rights and attributes of individuals (physical or logical, i.e. M2M). IAM objectives are to enforce the access policies, provide auditing, logging and reporting functionalities.

Identity is the digital representation of data relating to a specific individual, whether he/she is an employee, partner, supplier or customer. Identity management includes all the management functionalities related with controlling the roles and access entitlements of these individuals.

A typical IAM system have a number of components, usually with overlap functionalities. Among the most common are:

- Directory Services – handle the storage of critical data such as user profiles, accounts, password and entitlements, among others.
- Authentication – technologies and mechanisms to establish whether users are who they claim to be.
- Single Sign-On (SSO)1 - provides the procedures needed to guarantee that a user, after performing a single authentication, can access multiple systems.
- Identity lifecycle management – offers the functionalities to create and delete accounts, manage account and entitlement changes and track policy compliance.
- Identity Provisioning/deprovisioning – The automatic creation and expiration of accounts in multiple systems based on data from authoritative data sources, thereby reducing the administrative effort involved in manual account creation and management, and reducing security risk by the automatic application of policies.
- Workflow – The automation of steps within the identity lifecycle management process including notification, approval, escalation and creation of audit data.
- Credentials and Passwords Management – Passwords, certificates and smart cards management, both in a configuration application or by the user itself, using a self-service portal application.
- Authorization and Access Management – facilities the creation and maintenance of rules needed to establish that users are authorized to undertake particular actions (in a Role Based Access Control, this includes the role definition and role membership).
- Access Governance – to guarantee that the access grants are in line with internal or external security requirements.

2.2 Standardization and Initiative Groups

The Identity Management area, and all the related domains, has been the ground for several initiatives. Some of the initiatives are drive by more traditional organization groups, as ITU-T or ETSI, however an important amount of work have been proposed by ah-doc initiatives that in the end usually try to include the results in traditional standardization groups. OAuth protocol (nowadays an

[1] Single Sign-Off is SSO reverse process. It forces the invalidation of Service Provider's individual session and the SSO IAM session. Technically there are a couple of approaches, none of them standard or *bullet-proof*.

IETF RFC) followed such flow, was initiated as an ad-hoc work and moved to IETF when results its utilization got momentum.

http://kantarainitiative.org/

Kantara Initiative is an open focal point for collaboration to address the issues we each share across the identity community. Kantara activities focus on requirement gathering for the development and operation of Trust Frameworks as well verification of actors within Trust Framework ecosystems. Kantara Initiative Accredits Assessors, Approves Credential Service Providers Services and Recognizes Service Components (Identity Proofing and Credential Management). [2]. Working Groups (WG) within Kantara address different aspects that goes from Attribute Management to National Strategy for Trusted Identity in Cyberspace. [3]

https://www.oasis-open.org/

OASIS (Organization for the Advancement of Structured Information Standards) promotes industry consensus and produces worldwide standards for security, Cloud computing, SOA, Web services, the Smart Grid, electronic publishing, emergency management, and other areas. OASIS open standards offer the potential to lower cost, stimulate innovation, grow global markets, and protect the right of free choice of technology. [4]

OASIS defines a set of standards related with Identity Management topics. Among them are the following one:

- SAML (Secure Assertion Markup Language) protocol and bindings
- XACML (eXtensible Access Control Markup Language)
- SPML (Service Provisioning Markup Language)

http://www.etsi.org/WebSite/homepage.aspx

ETSI produces globally applicable standards for Information & Communications Technologies including fixed, mobile, radio, broadcast, internet, aeronautical and other areas [7].

ETSI ISG INS http://portal.etsi.org/portal/server.pt/community/INS/345

A couple of European Project's results were used to create an Industry Specification Group, within ETSI, named Identity Management for Networks and Services[2] . This group takes Identity and Access Management beyond the current prime focus on the web and application domains and some limited NGN (Next Generation Network) areas, to focus on networks and services in the future digital space. A major concern is to support interoperability as well as federation at all levels including networks, such as between operators, enterprise and home [8].

[2] Portugal Telecom was one of the four founders of this group, being active and leading one of the working items - IdM Interoperability between Operators or ISPs with Enterprise, since the beginning

The mission of the IETF is to make the Internet work better by producing high quality, relevant technical documents that influence the way people design, use, and manage the Internet [12].

OAuth – http://oauth.net/about/

In order for these applications to access user data on other sites, they ask for usernames and passwords. Not only does this require exposing user passwords to someone else – often the same passwords used for online banking and other sites – it also provides these application unlimited access to do as they wish. They can do anything, including changing the passwords and lock users out.

OAuth provides a method for users to grant third-party access to their resources without sharing their passwords. It also provides a way to grant limited access (in scope, duration, etc.). [10]

SCIM – http://www.simplecloud.info/

The System for Cross-domain Identity Management (SCIM) specification is designed to make managing user identities in cloud-based applications and services easier. The specification suite seeks to build upon experience with existing schemas and deployments, placing specific emphasis on simplicity of development and integration, while applying existing authentication, authorization, and privacy models. Its intent is to reduce the cost and complexity of user management operations by providing a common user schema and extension model, as well as binding documents to provide patterns for exchanging this schema using standard protocols. In essence: make it fast, cheap, and easy to move users in to, out of, and around the cloud [9].

www.tmforum.org

TM Forum is a global, non-profit industry association focused on enabling service provider agility and innovation. As an established thought-leader in service creation, management and delivery, the Forum serves as a unifying force across industries, enabling more than 900 member companies to solve critical business issues through access to a wealth of knowledge, intellectual capital and standards.

Enterprise Security Team (ESEC)
http://www.tmforum.org/EnterpriseSecurity/9934/home.html

Enterprise Security Management standards enable:

- User and identity management: An integrated view and efficient, automated management of system access based on user roles and entitlements, per user, per system (TMF615).
- Single sign-on/off: A controlled and automated sign-on/off standard providing access to multiple systems, including automated password management and control, based on business role (TMF614).
- Compliance reporting: Automated reporting and auditing of what the user has done, including who, which roles, when, and on what system. [13]

http://openid.net/connect/

OpenID Connect 1.0 is a simple identity layer on top of the OAuth 2.0 protocol. It allows Clients to verify the identity of the End-User based on the authentication performed by an Authorization Server, as well as to obtain basic profile information about the End-User in an interoperable

and RESTful manner [5].

2.3 Federation

Federated identity management is an arrangement that can be made among multiple enterprises that lets subscribers use the same identification data to obtain access to services of all enterprises in the group, but federation is more than just the ground to offer Single Sign-On.

Federation implies delegation of responsibilities, through trust relationships between federated parties. Authentication is just one form of delegated responsibility. Authorization, profile management, pseudonym services, and billing are other forms of identity-related functions that may be delegated to trusted parties.

Since federation implies that a responsibility (i.e. authentication) is delegated to and performed by a different party, a protocol may be in place that allows individuals to obtain an assured claim (essentially tamper-proof claims that a given identity has successfully completed an action or is entitled to a collection of privileges). For example, in the case of federated sign-on, an authenticated identity obtains an assured claim that proves that the individual has successfully authenticated with an approved authentication service.

Federation, implies trust and trust may be established in following different ways, for example Hub-and-spoke, Hierarchical, Peer-to-peer Web of Trust.

2.4 Authentication

Authentication is a process by which a system verifies the identity of a User who wishes to access it. Since Access Control is normally based on the identity of the User who requests access to a resource, Authentication is essential. Authentication may be implemented using Credentials, Smart Cards or even a Public Key Infrastructure.

The process normally consists of four steps:

1. The user makes a claim of identity, usually by providing a username.
2. The system challenges the user to prove his or her identity. The most common challenge is a request for a password.
3. The user responds to the challenge by providing the requested proof.
4. The system verifies that the user has provided acceptable proof by, for example, checking the password against a local password database or using a centralized authentication server.

2.4.1 SAML

Security Assertion Markup Language (SAML) [6] is an XML standard that allows secure web domains to exchange user authentication and authorization data. Using SAML, an online service provider can contact a separate online identity provider to authenticate users who are trying to access secure content.

Initially published as a version 1.0 standard in 2002, SAML evolved over the years to its current state, SAML 2.0, in 2005. The SAML standard is managed by the OASIS Security Services Technical Committee. SAML 2.0 is actually a combination of three predecessor identity federation standards:

SAML 1.1, ID-FF (Identity Federation Framework) 1.2, and Shibboleth. OASIS Security Services TC is working on SAML 2.1 specifications.

Since it is XML-based, SAML has extensibility, which makes it a very flexible standard. Two federation partners can choose to share whatever identity attributes they want in a SAML assertion (message) payload as long as those attributes can be represented in XML. This flexibility has even led to pieces of the SAML standard, such as the SAML assertion format, being incorporated into other standards such as WS-Federation.

Interoperability also gives SAML a clear advantage over proprietary SSO mechanisms, which require the Identity Provider (IdP) and Service Provider (SP) to both implement the same communication protocol. For an enterprise, proprietary SSO means each new connection potentially requires new and different software implementation. With SAML, a single SAML implementation can support SSO connections with many different federation partners. Some large organizations, particularly those who have already gone through the pain of supporting multiple proprietary SSO implementations, are now mandating the use of SAML for Internet SSO with Software-as-a-Service (SaaS) applications and other external service providers.

A brief overview of the SAML process is presented next and illustrated in Figure 3.

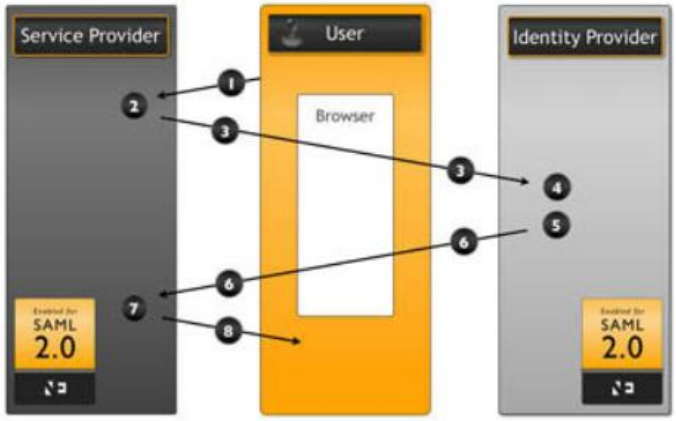

Figure 3 – SAML process

1. User requests a service from the Service Provider;
2. Service Provider verifies if the user has a SAML token;
3. User gets redirected to the Identity Provider (authentication delegation) ;
4. Identity Provider authenticates the user;
5. Identity Provider creates a SAML token;
6. User gets redirected to the Service Provider, with the SAML token;
7. Service Provider verifies the SAML token;
8. Service Provider gives the requested information to the user.

2.4.2 OpenID Connect

OpenID Connect is a suite of lightweight specifications that provide a framework for identity interactions via REST like APIs. The simplest deployment of OpenID Connect allows for clients of all types including browser-based, mobile, and javascript clients, to request and receive information about identities and currently authenticated sessions. The specification suite is extensible, allowing

participants to optionally also support encryption of identity data, discovery of the OpenID Provider, and advanced session management, including logout [5].

OpenID Connect Clients use the scope values as defined in OAuth 2.0 to specify what access privileges are requested for Access Tokens. The scopes associated with Access Tokens determine what resources will be available when they are used to access OAuth 2.0 protected endpoints. For OpenID Connect, there is the concept of UserInfo (represented by a specific Endpoint) that allows to request extra information about an user. The OpenID connect scopes vary from profile, email, to phone or address.

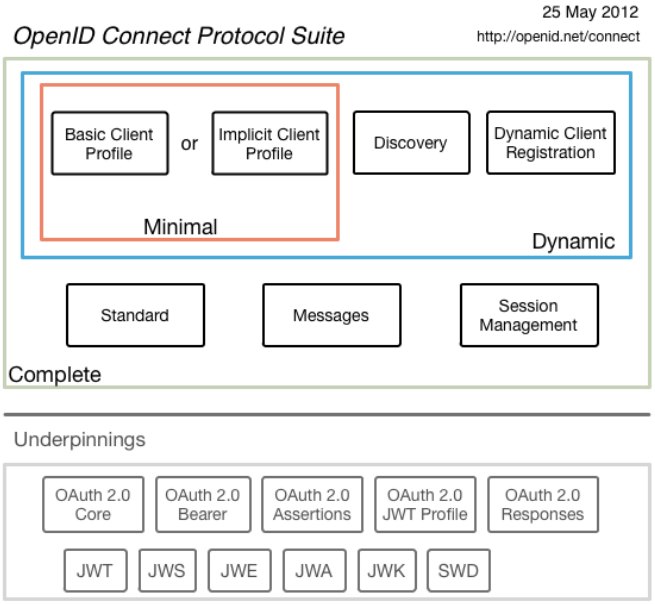

Figure 4 – OpenID Connect Protocol Suite [5]

The OpenID flow, for authorization, consists of the following steps (first set of interactions shown in Figure 5):

1. Client prepares an Authorization Request containing the desired request parameters (including the scope value);
2. Client sends a request to the Authorization Server;
3. Authorization Server authenticates the End-User;
4. Authorization Server obtains the End-User Consent/Authorization;
5. Authorization Server sends the End-User back to the Client with code;
6. Client sends the code to the Token Endpoint to receive an Access Token and ID Token in the response.

The Access Token as the same use as defined in OAuth 2.0 specification and the ID Token is a JSON Web Token (JWT). JWT is a compact URL-safe means of representing claims to be transferred between two parties. The claims in a JWT are encoded as a JavaScript Object Notation (JSON) object that is used as the payload of a JSON Web Signature (JWS) structure or as the plaintext of a JSON Web Encryption (JWE) structure, enabling the claims to be digitally signed or MACed and/or encrypted. This is one of the OpenID advantages when compared with the actual use of OAuth for authentication delegation.

Prior to the flow described previously, an enrollment procedure has to be guaranteed before. This process is similar to the one used in the OAuth protocol and results in the issue of an Application key and secret.

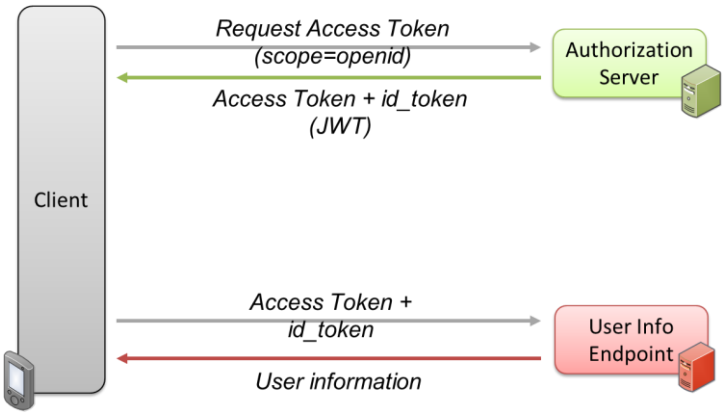

Figure 5 – OpenID Connect simplified message flow

When a client wants to access a certain resource, in the user's behalf, it sends the Access Token and the ID Token to the resource owner. The resource owner verifies (with the Authorization Server) whether both tokens are valid and, if they are, it provides the resource to the client.

2.5 Authorization

By definition authorization is the process of giving someone permission to do. Different authorization models and, at least, two granularity models are available in literature, when deal with authorization. The authorization models are presented in next sub-sections.

Authorization granularity is usually divided in fine-grained and coarse-grained access. Fine-grained access control works on smaller items whereas coarse-grained access control will work on larger items.

Granularity can apply to the message being intercepted or the information being considered for access control. Here after are presented some examples that evidence the difference between both of them.

- Coarse: Employees can open the door.
- Coarse: Users can access to a service (i.e. web application).
- Fine: Employees based in the Altice can open or close the door during office hours.
- Fine: Employees in the Engineering department and based in the Altice can open or close the door during office hours if they are assigned to an active project. [14]
- Fine: User can view the report X in view Y on service Z.
- Altice Labs Identity and Access Manager supports both authorization granularities, depending on the configuration.

2.6 Authorization models

Different authorization models are available in literature. Grossly speaking three types of authorization models have to be taken into consideration, due to its widely use.

2.6.1 Access control lists

With Access Control Lists (ACL), once a user is authenticated, the user is allowed to access an application or not depending on whether that user's id is on a list of authorized users (white list) or blocked users (black list). This mode is either all-in or all-out. It is extremely coarse-grained from that perspective. It is also coarse-grained in the sense that it only considers one dimension, that of the user, ignoring resources, actions, and context. In the previous examples, John is the user id that would be on the ACL for the open door action [14].

2.6.2 Role-based access control

In the Role-based Access Control (RBAC), it is not who you are but rather what role(s) you embody. For instance, a person might be allowed to open a door. That right (or permission) is granted because that person has the role employee, not because of who they are. Typically, in RBAC, a user can have multiple roles to which different permissions can be granted. Let's take a simple example. The popular blogging platform - WordPress. WordPress lets users define roles and associate permissions to roles and then roles to users therefore transitively granting users permissions. But roles have their limits too. How do you express other conditions or parameters? What if you want to express a permission of the following form: "only employees can open the door between 9AM and 5PM". With RBAC, you can express employee open door role. But it is not possible (or difficult) to really express the time constraint. In addition, what if you want employees who can only close doors? You would need to define employee open and an employee close role. This leads to role explosion [14].

2.6.3 Attribute-based access control

Instead of defining permissions based on roles only, one should be able to use attributes. This is Attribute Based Access Control (ABAC). Attributes are any bit of data, or label that describes a user, resource, target, object, environment, or action. Anything from an apple, its color, and weight to a person eating that apple, the location of the apple… With ABAC, you can mix and match attributes to define extremely targeted (fine-grained) rules e.g. employees can open the door between 9AM and 10AM or close the door after 5PM if and only if the employee belongs to the "door opener" group [14].

Altice Labs IAM provides the flexibility for the applications to use both models - RBAC and ABAC, depending on the IAM configuration.

2.6.4 OAuth

OAuth is an IETF standard for authorization. OAuth provides a method for clients to access server resources on behalf of a resource owner (such as a different client or an end-user). It also provides a process for end-users to authorize third-party access to their server resources without sharing their credentials (typically, a username and password pair), using user-agent redirections. Two different versions (1.0a and 2.0) of OAuth are nowdays disseminated and supported by most of the Web 2.0 providers (i.e.Facebook, Google, Twitter, Linkedin, etc.).

The protocol specifies four entities:

- Authorization Server – The server that issues and validates the tokens. The tokens are the information that correlates the user, the requester and the resource scope authorized by the user.

- Resource Owner – The resource owner is the user that owns the resource protected by the resource server and that wants to be accessed by the Client/Application.
- Resource Server – The entity that stores the resource and provides it to the Client/Application after token validation in the Authorization Server.
- Client/Application – The entity that wants to access the resource owned by the user (resource owner).

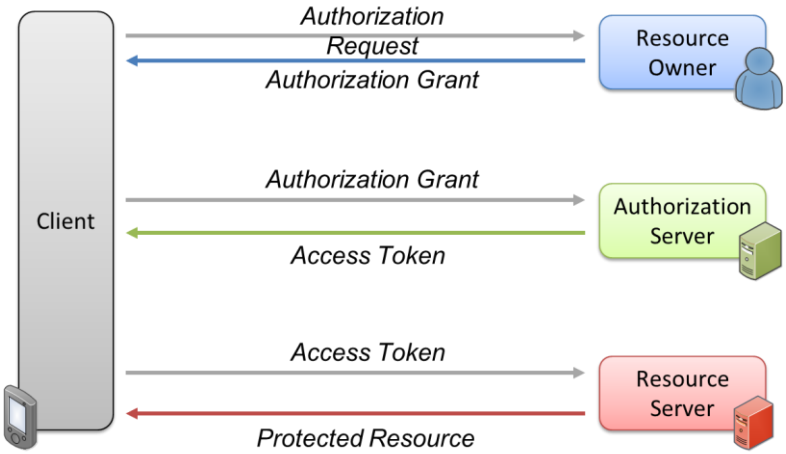

Figure 6 – Oauth 2.0 Flow overview

An enrollment procedure has to be guaranteed before the normal OAuth flow. The Client/Application has to be registered in the Authorization Server. The result of this enrolment is that the Client gets a unique API Key and API Secret, used to set-up a secure (i.e. HTTPS) transport channel, in order to guarantee that the application is authenticated, is identified and as authorization to access the authorization end points. Previous figure presents the OAuth 2.0 flow, which is very similar to the one presented in OpenID Connect section of this document.

2.6.5 XACML

The eXtensible Access Control Markup Language (XACML [19]) is an XML-based language, used to represent policies and to define how to evaluate them. It has been defined by the OASIS standardization group and, nowadays, it is the most accepted policy language.

XACML defines a flexible architecture that can be adapted to different scenarios and policy types. It defines the following main elements:

- Policy Enforcement Point (PEP) Controls the access to the resource and enforces the decision taken by the PDP
- Context Handler (CH) Obtains context information (e.g. user attributes) and requests a decision to the PDP
- Policy Decision Point (PDP) Evaluates the policies and takes a decision based on the available information from the CH
- Policy Information Point (PIP) Provides attribute values to the CH
- Policy Administration Point (PAP) Creates and manages the policies to be used by the PDP

Figure 7 shows an XACML simplified interaction, depicting the main elements of the XACML architecture and how they interacts to process an access request from a Requester entity.

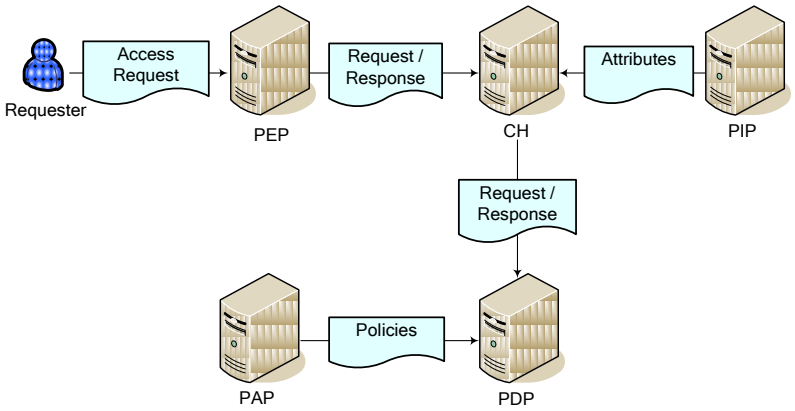

Figure 7 – XACML simplified interaction

In the first place, the Requester sends an Access Request to the PEP. The PEP determines that a decision needs to be taken; hence it sends a request to the CH. This request is specified using an application-dependent format between the PEP and the CH. The CH obtains the required information about the subject, resource, action and environment from the request, though it may obtain additional information from the PIP if required. Once this information has been gathered, the CH sends a request to the PDP, which uses this information to select the applicable policies from those available in the PAP. This request is done using a canonical format defined by XACML. Following, the PDP evaluates the policies and returns to the CH a response containing the taken decision. This decision is translated by the CH to the application-dependent format and sent to the PEP in order to be enforced.

2.6.5.1 XACML data model

XACML defines a data model to represent security policies in a clear and unambiguous language. This data model is depicted in Figure 8.

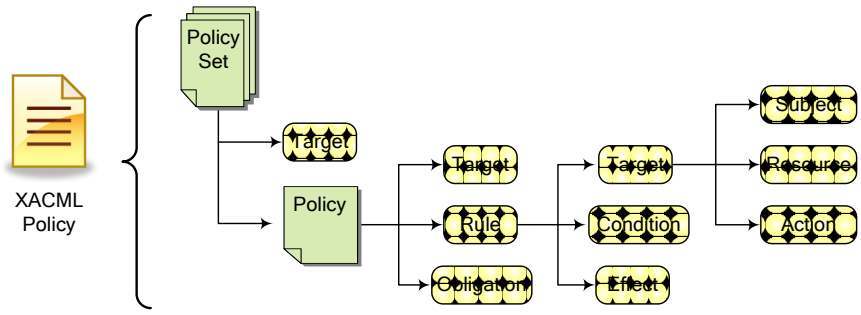

Figure 8 – XACML policy data model

The elements that compose the XACML Policy data model are the following:

- PolicySet. A PolicySet set is an aggregation of various Policies or PolicySets. The result from evaluation of different policies must be intelligently combined to yield a well-defined result. That is done using Policy Combining Algorithms.

- Policy. Consists on a Target, a set of Rules and, optionally, a set of Obligations. The Rules defined in a Policy are evaluated together to take an authorization decision, using a Rule Combining Algorithm to combine their output and to construct a unified decision. There exist different combining algorithms defined in XACML. For example the algorithm deny overrides, which states that if there is some rule that evaluates to deny, then the Policy is evaluated as deny. Another example, first applicable, states that each Rule has to be evaluated in the same order as it was defined in the Policy. If the Rule is applicable, the result is the effect of such a Rule.
- Target. This element contains a simplified set of conditions for the Subject, Resource and Action elements that are used to determine if this Policy (or PolicySet) is applicable to a given request.
- Rule. Rules are used to identify various conditions or cases under which a policy may become applicable. A Rule is composed by a Target, a Condition and an Effect.
- Condition. A Condition is a Boolean expression that is used to define the applicability of a Rule. It can be evaluate to True, False or Indeterminate.
- Effect. The Effect is the intended consequence of a satisfied Rule. It can be either Permit or Deny.
- Obligation. An Obligation is an operation specified in a Policy that should be performed in conjunction with the enforcement of an authorization decision.

Attributes are an important part of XACML. They are used to specify the characteristics of a subject, resource, action and environment. Then, Rules evaluates these attributes in order to determine whether some restriction is applicable or not. Attributes are identified by the *SubjectAttributeDesignator*, *ResourceAttributeDesignator*, *ActionAttributeDesignator* and *EnviromnentAttributeDesignator* elements. These elements use the *AttributeValue* element to define the requested value of a particular attribute. Additionally, the *AttributeSelector* element can be used to specify where to retrieve a particular attribute.

2.7 User's Management

2.7.1 SPML

The Service Provisioning Markup Language (SPML) is the open standard for the integration and interoperation of service provisioning requests. SPML is an OASIS standard based on the concepts of Directory Service Markup Language. The SAML protocol is used to exchange the provisioning messages between different domains [19].

SPML's goal is to allow organizations to securely and quickly set-up user's provisioning for Web services and applications, by letting enterprise platforms such as Web portals, application servers, and service centers generate provisioning requests within and across organizations. This can lead to automation of user or system access and entitlement rights to electronic services across diverse IT infrastructures, so that customers are not locked into proprietary solutions. These messages can be requests to add, modify, or delete user accounts, enable or disable access, grant or revoke access rights, change passwords, and all other types of provisioning tasks. By using SPML, heterogeneous systems can easily participate in provisioning business processes without needing complex and expensive integration. For example, a supply partner (Company A) goes to its partner's (Company B) supply chain portal and requests access to its inventory data, which is stored in a back-office system. In response, Company B initiates a request using SPML to communicate with SPML-enabled identity management software. After automatically acquiring the appropriate permissions, Company B grants

the appropriate access levels to Company A to gain access to the data it needs. This process takes place without the need for the portal environment to have an intimate understanding of the back-office environment, which brings a lot of automation to the provisioning process.

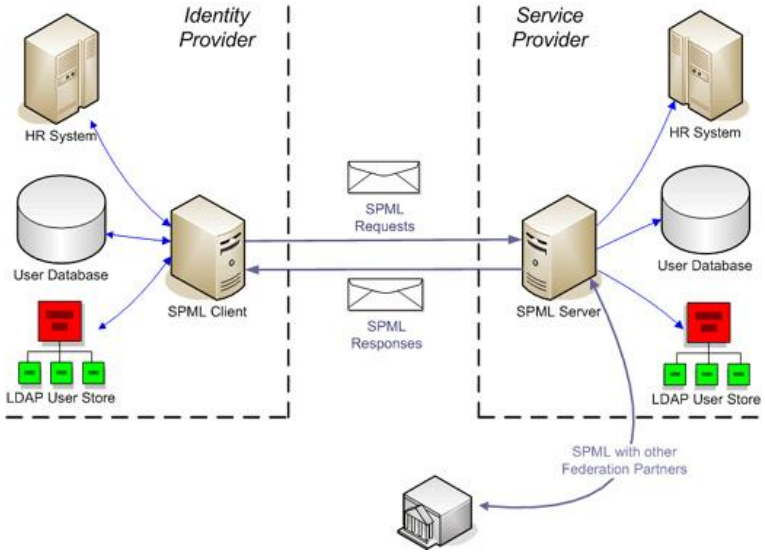

Figure 9 – SPML interactions overview

SPML is extremely useful to federated networks/services, because it allows the enterprises participating in the federation to exchange provisioning messages.

2.7.2 SCIM

The System for Cross-domain Identity Management (SCIM) is designed to make managing user identities in cloud-based applications and services easier. The specification suite seeks to build upon experience with existing schemas and deployments, placing specific emphasis on simplicity of development and integration, while applying existing authentication, authorization, and privacy models. Its intent is to reduce the cost and complexity of user management operations by providing a common user schema and extension model, as well as binding documents to provide patterns for exchanging this schema using standard protocols. In essence: make it fast, cheap, and easy to move users in to, out of, and around the cloud [20]. All the information exchanged is encoding in JSON, even if XML is also allowed and specified within the specification. For manipulation of resources, SCIM provides a REST API with a rich but simple set of operations, which support everything from patching a specific attribute on a specific user to doing massive bulk updates.

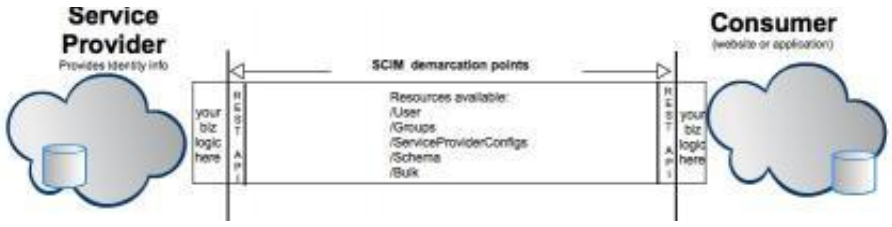

Figure 10 – SCIM overview

3. Altice Labs Identity and Access Manager

3.1 Introduction

Altice Labs IAM is a modular trustworthy framework that ensures the cut down of the expenses related with the identity and entitlement user's management in the today complex and heterogeneous enterprise environment.

Next subsections provide an overview the IAM functional characteristics.

3.2 Architecture

IAM follows a modular architecture, composed by different modules (shown in Figure 11), that can be deployed in different ways, in order to answer to specific scenario requirements.

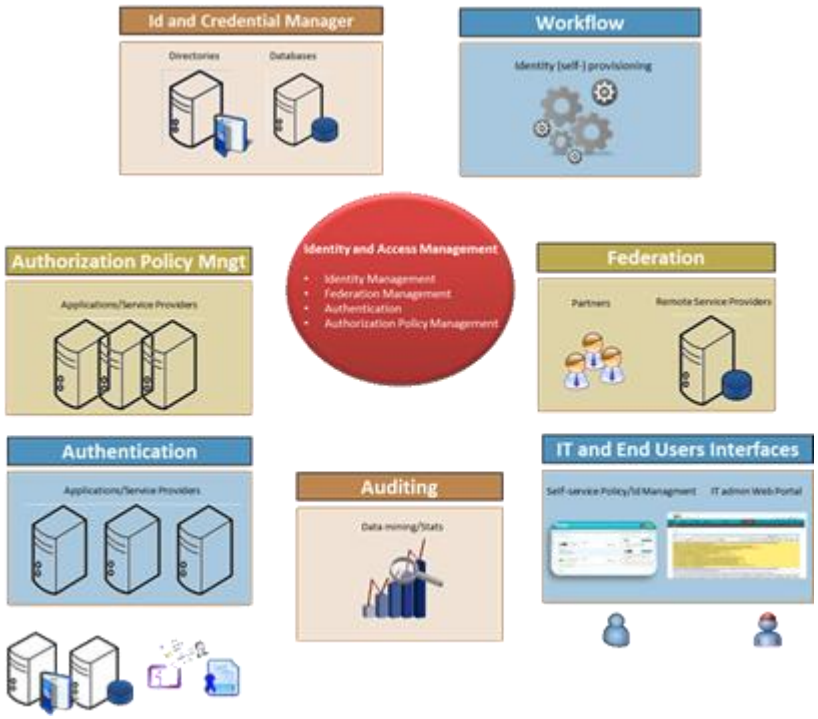

Figure 11 – IAM functional architecture overview

Each of the modules are described hereafter.

3.2.1 User's identity and attribute management

This module provides custom connections to many database systems (i.e. Oracle, Postgresql) and directory data repositories (i.e. LDAP, AD, Alfresco). IAM doesn't provide a centralized repository where the entire and distributed identity related information (user attributes, entitlements, etc.) is replicated and maintained. IAM keeps pointers to the user's information, stored in the distributed

stores. The concept of meta-user is used to create an instance of the user within IAM and to associate this new identity to the ones owned by the user in the different stores (his Accounts). This concept maps the user's identities with his representation in the distributed stores.

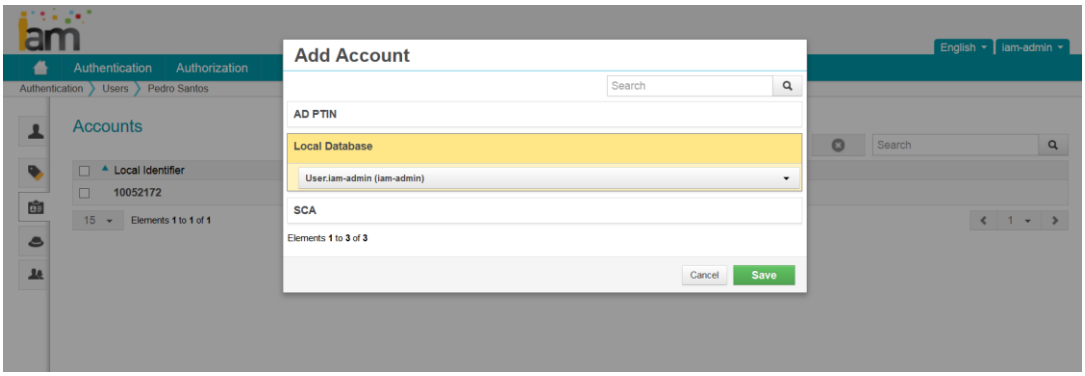

Figure 12 – *Meta-user* configuration

Figure 12 shows user *Pedro Santos* who exists in three different stores with a specific identifier in each store. The figure presents the user's identity (10052172) in the store named "AD Altice Labs". With this configuration the IAM knows that the user has attributes in different stores and is able to collect them, from those stores, on demand – i.e. when a certain application requests it.

Side by side with the user's configuration the IAM needs to keep track of the user's attributes used in the different repositories. This is essential to guarantee that the information is collected in the right way.

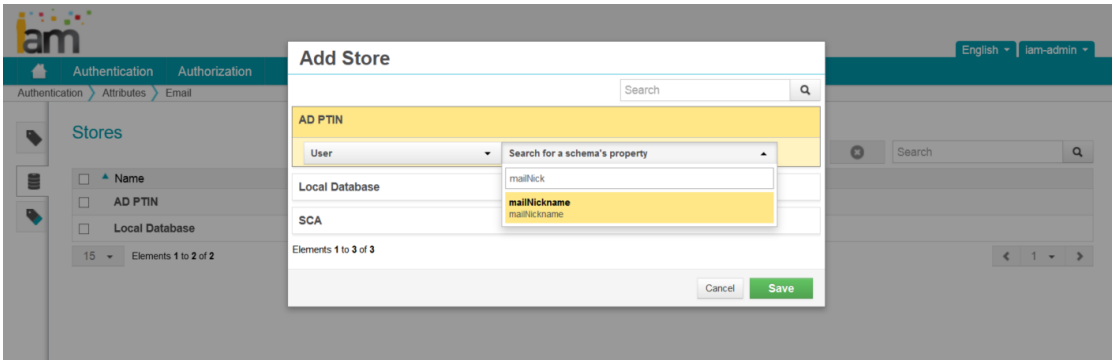

Figure 13 – *Meta-attribute* configuration

Figure 13 presents an example of the IAM attribute schema configuration. As in the user notion, IAM uses the concept of *meta-attribute* to create the "virtual" attribute schema that is composed by fields from different information stores/sources. In this example the attribute "Email" represents the field "mailNickName" in the "AD Altice Labs" repository (store), and by other fields in the other configured stores. Each service or application that desires to receive the user's email address it uses the *meta-attribute* name to identify the attribute. The IAM, based on its own configuration collects the field form the right store and sends the value to the requester:

- If the User has Priority Meta-Attributes Stores configured (an overwrite at User's level exists) them Claims are read according to that.

- Else, If the Service defines for each claim a store, claims are read from there (Overwrite at Service's level).
- Else, If the users has more than one account, claims are read from the account with lower priority.
- Else, Claims are read from the 1st user's account that IAM find.

Attributes can be named at several levels. When specifying a new attribute (and mapping it to the various information sources), the used name should be simple, short and should make sense within the organization's vocabulary, while for the presentation name, a name that makes the attribute unique and ties it to the supplying organization. In example showed previously, the presentation name should be "http://iam.alticelabs.com/email" marking the attribute as the "email" supplied by "alticelabs.com" "iam" system.

3.2.2 Federation

This module allows the establishment of trusted relationships with external partners or systems, allowing for the secure and interoperable exchange of attribute-based identity data (SAML, ID-FF) - Figure 14. Thereby it enables features such as Single Sign On/Off, role translation between domains or access rules based on point of origin.

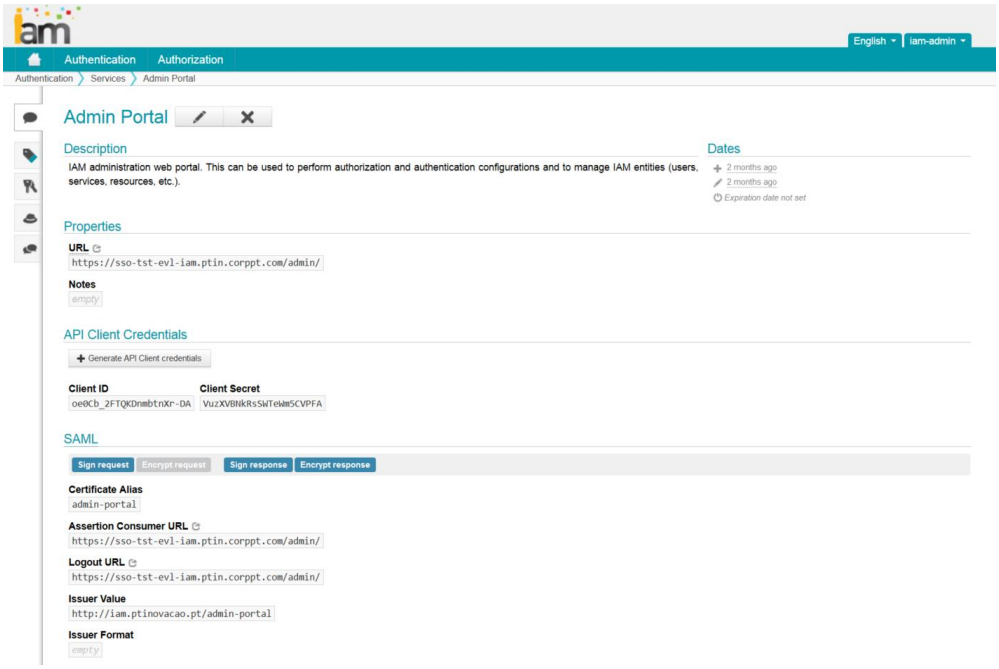

Figure 14 – Federation configuration

Moreover this module provides identity mapping functionalities to the end-user. With this the end-users is able to map their distributed identities on the demand. The mapping can be done based on the enterprise internal distributed stores – the meta-user concept applies (see previous section) – or it could be based on Web 2.0 services (i.e. Facebook). IAM's Federation component can work as an OAuth client. OAuth 1.0a and OAuth 2.0 are supported.

3.2.3 IT and User Interfaces

IAM has three different graphical user interfaces developed on top of web technologies as CSS3, HTML 5 and Javascript.

1. Self-service portal – in this portal the end-user is able to manage his identity in multiple domains. It is possible for the end-user to map his identities on demand. He can check in which application he is logged and force the logout per application. Moreover it is possible to change some user's attributes, for example its password or a set of extra attributes permitted by the administrator.

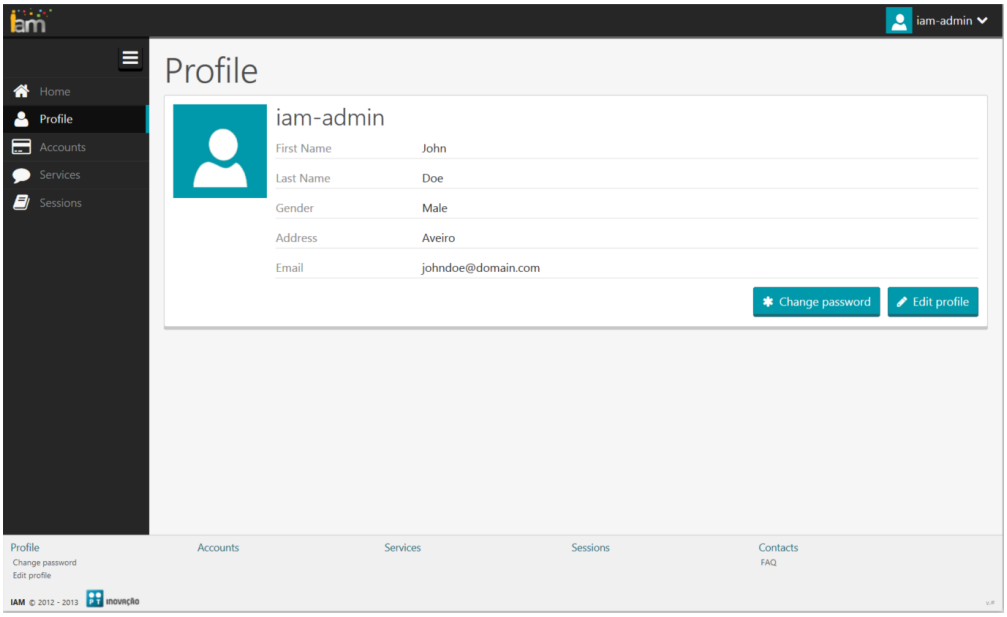

Figure 15 – IAM self-service portal

2. Admin Portal – The Admin Portal provides the mechanisms for IT administrators to easily configure the entire system. It is the unique point of contact – console – for system administration. Services, users, authorization and auditing and reporting are some of the functionalities available for the IAM administrators.

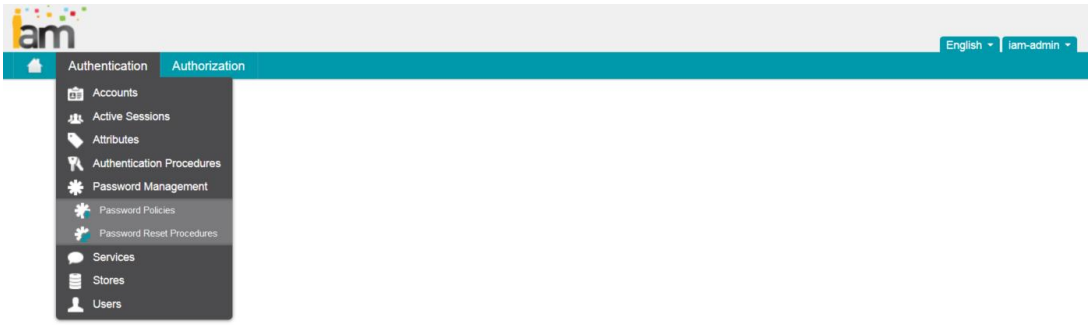

Figure 16 – IAM Admin Portal portal

3. Authentication Portal – The authentication portal is the portal responsible to gather the user's credentials. This portal is configurable in terms of elements and layout, once it will depend on the client it is deployed.

Figure 17 – Authentication Portal

3.2.4 Auditing & Workflow

IAM provides logging capabilities, report generation, statistical analysis and configurable alarm conditions. Definition of workflows regarding identity provisioning and access authorization is available. With this, clients have the possibility to configure IAM to fit its own business needs.

3.2.5 Authentication

This module facilitates intra/inter-domain authentication by providing a single authentication interface supporting multiple authentication procedures (OpenID connect, username/password, smartcard-based, X.509). The authentication procedures is based on a plugin architecture, which means it is possible to include more authentication types on *demand* in a very simple way. The authentication module works as an Authentication Server or a Secure Service Token (STS).

As said previously, IAM does not demand for the use of a centralized database with all the users and their attributes (including passwords). The authentication is always performed "against" one of the distributed stores where the user exists (his Accounts). The choice is based on the configuration of the service the user is trying to access.

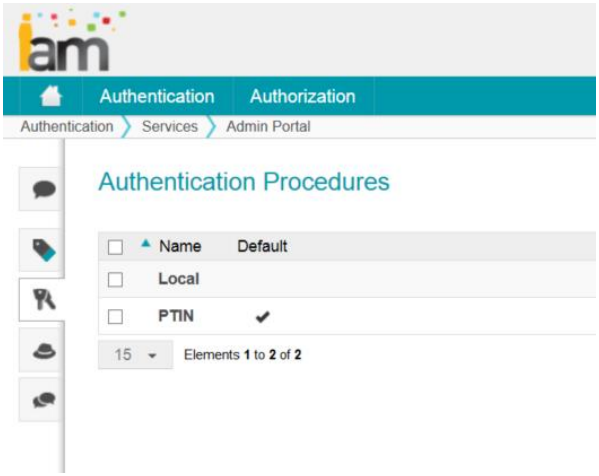

Figure 18 – Service configuration

Figure 18 presents the configuration of a service, it shows that for that particular service the authentication should be done in the “Altice Labs” store, nevertheless it’s possible to select “Local” store. This happens for all the users that tries to authenticate on the service.

The authentication server is based on SAML (see section 3.4.1), it implements the HTTP POST bindings. While not mandatory by the binding’s specification, IAM requires that all clients use HTTPS communication. This requirement allows the server to turn the token encryption on or off without creating extra security concerns.

3.2.5.1 Authentication Proxy

For services that cannot delegate authentication, IAM provides an Authentication Proxy. This Proxy intercepts authentication request to web applications, redirects to IAM where authentication is done. Users credentials used in IAM can be the same or others from those used in Service Provider.

1. User requests a service from the Service Provider;
2. Proxy validates if Service Provider shows login page;
3. If so, user gets redirected to IAM sending a SAML Request;
4. IAM validates if there is an active session. If there isn’t, IAM’s login page is shown;
5. User submits credentials;
6. IAM answers the SAML Request, sending an OK or NOK. If Ok, an access token to IAM’s API is sent;
7. Proxy invoke IAM’s API using token as parameter. If the token is valid, user’s credentials are returned;
8. Proxy submits credentials in Service Provider authentication form;
9. Identity Provider authenticates the user.

3.2.6 Authorization

An application main focus should be on its core business requirements. In most applications, authorization, although an essential requirement, is usually not part of the core business requirements. In fact, more often than not, authorization is “added” to an application as an afterthought. Usually these application-specific authorization models are tightly coupled to the business models. As business models change (as they often do over time) extra effort is need to extend, adapt and sometimes re-do the corresponding authorization model. Effort that is taken away for core business requirements. Other downsides include:

- When security policies change, all changes need to go through lengthy development and test cycles;
- Inability to rapidly respond to threats and security breaches;
- It is difficult to analyze and audit security policies and runtime authorization decisions;
- Application developers are forced to reinvent the wheel over and over, leading to longer development cycles, higher cost, and often rigid or lacking security implementations.

In order to relieve applications from designing their own authorization models and implementing their own authorization engines IAM provides a unified authorization model and authorization query API, enabling applications to focus almost exclusively on their core business requirements.

3.2.6.1 Model

IAM uses a hierarchical role based access control (RBAC) to define authorization policies and then expose those policies to properly authorized 3rd parties.

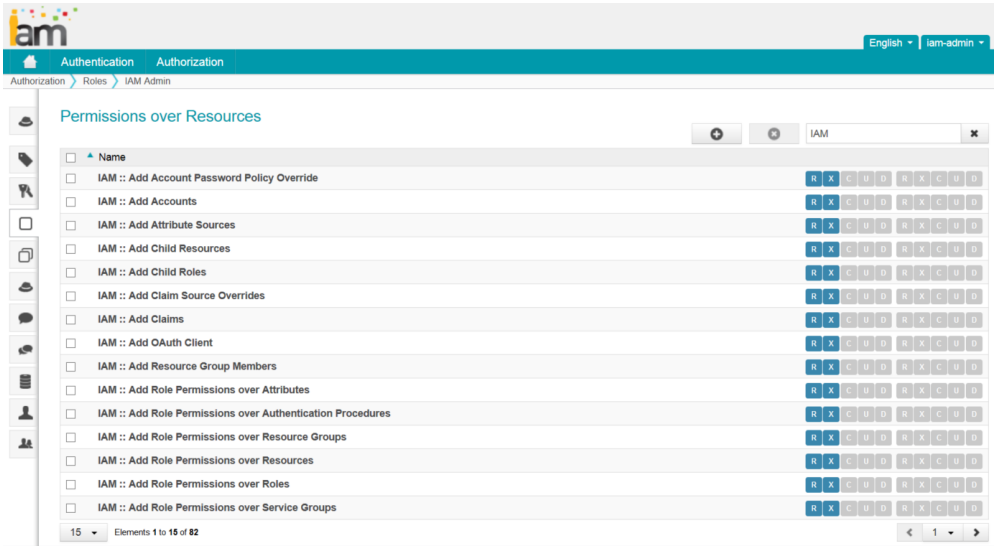

Figure 19 – Role permissions

These policies consist of what roles users or services have, and what permissions those roles grant over which resources (or users, or services or roles).

Entity	Description
User	The end-user, usually a person, requesting access to a Resource. The User has no access directly as a User, only in a Role.
Service	Web-portal, mobile app, daemon, or any other application/service (IAM API client). Service can be seen as a set of Resources. Typically a Service is an Application that wants/needs to implement access control. In architecture PEP/PDP, a Service is a PEP.
Resource	A representation of a service domain concept or entity (a web page, a button, a physical door, an action, etc.) that a User may need to access and the service check point for access control.
Role	Comprised of a set of permissions given over resources, users or services.
Permission	A pre-defined set of grants (**create**, **read**, **update**, **delete** and **execute**) which may be given to a role over a user, resource, service or another role.

Table 1 – Some Entities of IAM Authorization Model

Although authorization policies are configured per service and per user (i.e. roles are given to users or services), user policies are never evaluated by themselves. Any and all decisions regarding what a

user can or cannot perform is done within a service context. Thus an authorization decision (concerning a user action) is always made considering what that user and the service in question are allowed (or not) to do. The same does not apply to services since they can perform actions by themselves and therefore, authorization decisions are made based on only those services permissions.

3.2.6.2 Groups

Users, services and resources can be further organized in user groups, service groups and resource groups respectively, and like in respect to individual elements, roles can be given permissions over these groups. When that happens, permissions granted over a group apply to all group members (as long as they are members of the group in question).

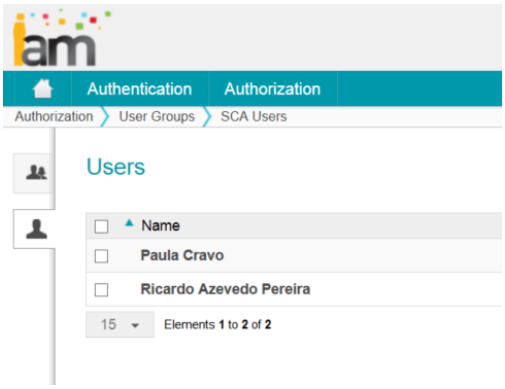

Figure 20 – Members of a user group

3.2.6.3 Hierarchies

Both roles and resources can be structured in hierarchies. Resource hierarchies are strictly for organizational purposes (they don't affect authorization). Roles hierarchies on the other hand do affect authorization as a role inherits all its children permission grants.

In both resource and role hierarchies, any one node can only have one parent but can have as many children as desired.

3.2.6.4 Permissions

IAM defines 5 permissions: create, read, update, delete and execute. These permissions can be granted to roles over resources, users, services, and corresponding groups. However resources and resource groups represent a special case.

Since resources represent a service domain concepts but at the same time are also an IAM entity, when granting a role permissions over a resource one needs to specify whether those permissions apply to the service domain concept (what that resource represents), or if they apply to the resource itself (as an IAM entity).

Consider for example a resource called "table entry" which a service uses to control the users that can "create" or "delete" table entries from some table. There's a semantic difference between being able to delete a table entry and deleting the resource "table entry" from IAM.

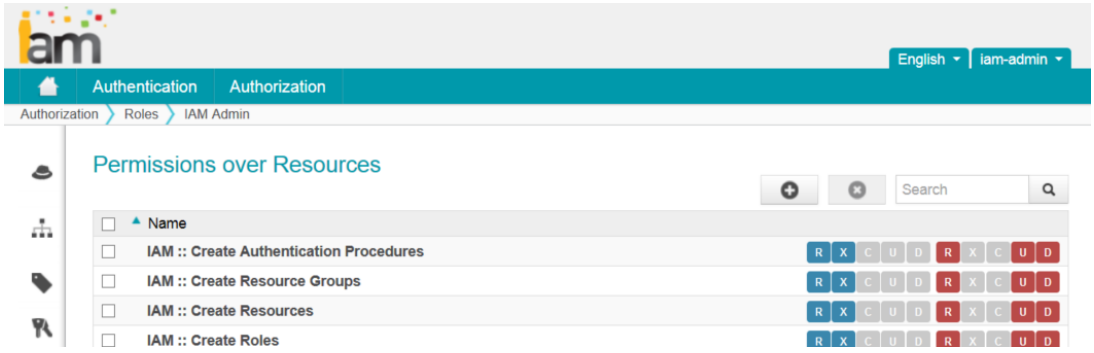

Figure 21 – IAM and Service Permissions

In order to clearly separate permissions over resources and permissions over what those resources represent, IAM defines two categories (or sets) of permissions: "iam" permissions (in blue in the Figure 21) and "service" permissions (in red in the same picture). The former can be granted (to roles) over any IAM entity (including resources and resource groups) and represents the permissions over the entity itself. The latter can only be granted to resources and resource groups and represent the permissions over what those resources or resource groups represent within the service domain. The semantics of "iam" permissions are therefore well defined (read means able to view, delete able to delete, etc.) the same is not true for "service" permissions. It is up to the services to define appropriate semantics to these permissions, as they apply in the domain in question.

3.2.6.5 Authorization Mechanics

As was previously said, access to IAM's API can be done in one of two "contexts": a "user/service" context and a "service" context. These are realized in the form of OAuth 2.0 tokens and are a result of an authentication procedure. OAuth tokens can be obtained by:

- User authentication at IAM's authentication portal
- OAuth 2.0 resource owner credentials grant flow
- OAuth 2.0 client credentials grant flow

In the first two cases the issued OAuth token represents a "user/service" context (i.e. a user using a specific service), while the last represents a "service" context.

For every API call, IAM acts as both a policy decision point (PDP) and policy enforcement point (PEP) which means the invoking party (user/service or service) will only be able to perform actions that its roles allow. However, in the most common authorization scenario, where a service uses IAM to configure authorization policies, the service itself will also act as a PEP as it will enforce authorization decisions based on information retrieved from IAM.

The way a service uses IAM to define the policies that it will eventually enforce is very flexible and can vary from service to service. For some services the simple presence of a role associated with a user will be enough to enact an authorization decision. For others, role hierarchies and resource can be necessary to model their application domain.

IAM doesn't answer binary questions (i.e.: "does user X have read access to Y?") but instead provides services with entitlements information (i.e. "to which resources does the user X have read access to?" or "which permissions does the user X have over the resource Y?") and it is up to the service to process and apply that information as it sees fit.

3.2.6.6 Disabled entities

Users, roles, resources, service, etc. can become disabled, either manually or through some automated process (like reaching the expiration date for example). Once disabled, that entity ceases to "exist" for an authorization procedures. What this means is that if, for example, a role is given to a user and is afterwards disabled, for authorization purposes, it's as if that user didn't have the role in question.

3.2.7 REST Interface

The entire set of Altice Labs IAM functionalities are exposed by REST services. REST defines an architectural style based on a set of constraints for building things the "Web" way. REST is not tied to any particular technology or platform – it's simply a way to design things to work like the Web. Altice Labs IAM follows this philosophy.

Table 1**Error! Reference source not found.** describes the HTTP methods, used by IAM, and their semantics.

Method	Description
GET	Requests a specific representation of a resource
DELETE	Deletes the specified resource
POST	Submits data to be processed by the identified resource

Table 2 – IAM HTTP methods

If the returned status code is higher than 400, then some error occurred. More information about the error is given in the message body in JSON format.

Status Range	Description
200	Successful
201	Created
400	Bad request
404	Not Found
409	Conflit
422	Unprocessable entity
500	Internal server error
501	Not Implemented

Table 3 – IAM WS status codes

IAM REST services are divided in two groups, functional and operational. The functional includes all the services that permits applications to manage, CRUD operations, in the internal IAM model (i.e. create a resource or a user). The operational group includes all the services that applications may use to perform IAM core functionalities, for example authenticate a user or verify if a user is able to operate over a resource.

Application and/or user authentication, filtering, sorting and pagination are available in all the services. To each application is offered an API key and a secret that are used to establish the communication channel (HTTPS using Basic Authentication) and to identified and authorize the application when accessing the REST services.

4. Conclusion

Embedding authentication and or authorization decisions in application code leads to static policies which cannot keep pace with changing security requirements. Not having a centralized policy management and uniform enforcement infrastructure leads to security silos where each application uses a different security mechanism and the resulting authorization policies cannot be reused across organizations.

On the other side actual enterprises are facing strong difficulties due to the existing of identity silos, where user's identity (basically credentials and attributes) are closed in different, and distributed, information stores/sources. The heterogeneity and complexity of nowadays enterprises doesn't permit Single Sign-On or the ability to cope with security standardization or auditing trails. The existence of a central Identity and Access Management system is essential for all organizations.

Altice Labs IAM provides a set of essential functionalities to address the mentioned problems of today's enterprises and to facilitate the application's development in terms of security and access rights (entitlements). More than just a unique product that addresses all the problems, IAM is a trustworthy framework that developers can use to leverage their applications in terms of security, and can be deployed separately in order to address the effective needs of today's enterprises. IAM may play the role of an Authorization Server only or an Authentication Server, depending on the client needs.

5. References

[1] https://developers.google.com/google-apps/sso/saml_reference_implementation

[2] http://kantarainitiative.org/

[3] http://kantarainitiative.org/groups/

[4] https://www.oasis-open.org/org

[5] http://openid.net/connect/

[6] https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=security

[7] http://www.etsi.org/WebSite/homepage.aspx

[8] http://www.etsi.org/WebSite/document/Technologies/LEAFLETS/Identity%20and%20access%20management%20for%20Networks%20and%20Services_2010_02.pdf

[9] http://www.simplecloud.info/

[10] http://oauth.net/

[11] www.tmforum.org

[12] http://www.ietf.org/

[13] http://www.tmforum.org/EnterpriseSecurity/9934/home.html

[14] http://www.webfarmr.eu/2011/05/coarse-grained-vs-fine-grained-access-control-part-i/, December 2012

[15] http://hitachi-id.com/password-manager/docs/defining-enterprise-identity-management.html, December 2012

[16] http://www.securityfocus.com/columnists/322, December 2012

[17] *Enterprise Identity Management: It's About the Business*, Jamie Lewis, The Burton Group Directory and Security Strategies Report, v1 July 2nd 2003

[18] M. Jones, et al, JSON Web Token (JWT), IETF OAuth Working Group, Internet-Draft 27/12/2012

[19] OASIS eXtensible Acces Control Markup Language TC. http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=xacml

[20] http://www.simplecloud.info/#overview, December 2012

6. Acronyms

AAA	Authentication, Authorization, Accounting
ACL	Access Control List
ABAC	Attribute Base Access Control
HTTP	Hyper Text Transfer Protocol
ID	Identity
IdM	Identity Management
IdP	Identity Provider
IETF	The Internet Engineering Task Force
RBAC	Role Base Access Control
RFC	Request For Comment
SAML	Secure Assertion Markup Language
SAML	Secure Assertion Markup Language
SaaS	Software as a Service
SCIM	System for Cross-domain Identity Management
SPML	Service Provisioning Markup Language
SSO	Single Sign-On(ff)
SP	Service Provider
RP	Relying Party
XACML	eXtensible Access Control Management Language
XACML	eXtensible Authorization Control Markup Language
XML	eXtensible Markup Language
M2M	Machine-to-Machine
TMForum	Telecommunications Management Forum

Rua Eng. José Ferreira Pinto Basto
3810-106 Aveiro
Portugal

Tel.: +351 234 403 200
Fax: +351 234 424 723

www.alticelabs.com