Interface Gráfica Web para Gerência de Usuários e Permissões de Acesso na Plataforma FIWARE

Irene Ginani Costa Pinheiro[1], **Carlos Eduardo da Silva**[1]

[1]Instituto Metrópole Digital - Universidade Federal do Rio Grande do Norte (UFRN)
Avenida Senador Salgado Filho, 3000, Lagoa Nova – CEP: 59.078-970 – Natal/RN

ireneginani@gmail.com, kaduardo@imd.ufrn.br

Abstract. *By working on some projects that involve the FIWARE Identity Provider, we noticed that there are some issues with the management of users, roles, and permissions. Although it provides a graphical Web interface, several functionalities are unsupported, such as editing user accounts or hierarchical RBAC implementation, for example. Thus, this work aims to present a graphical Web interface to simplify the manipulation of user accounts and access control policies for FIWARE platform.*

Resumo. *Ao trabalhar em alguns projetos que envolvem o provedor de identidades da plataforma FIWARE, percebemos que este possui problemas em relação a gerência de usuários, papéis e permissões. Embora ofereça uma interface gráfica web, diversas funcionalidades não são suportadas, como edição de contas de usuários ou implementação do RBAC hierárquico. Assim, este trabalho visa apresentar uma interface gráfica web para simplificar a manipulação de contas de usuário e políticas de controle de acesso.*

1. Introdução

O propósito do controle de acesso é limitar as ações dos seus usuários, de forma que informações, aplicativos, ou softwares sejam acessados somente por aqueles autorizados para tal [Sandhu and Samarati 1994]. Para isso, utilizamos várias ferramentas que auxiliam a utilizar na prática esse conceito. Dentre elas, temos a plataforma FIWARE[1], que foi projetado com uma abordagem *secure by design*, onde atributos de segurança da plataforma, e das aplicações que executem sobre ela, são considerados através de uma arquitetura que engloba questões relacionadas a gestão de identidades, controle de acesso e monitoramento de eventos de segurança. [García Vázquez et al. 2011]. Isso é conseguido com um conjunto de *Generic Enablers* (GEs), ou habilitadores genéricos, que facilitam a implementação de aplicações, e podem ser integrados com outros GEs da plataforma [Andriani et al. 2015, Di Cola et al. 2015, Moltchanov and Rocha 2014].

No contexto do projeto SmartMetropolis[2] utilizamos a plataforma FIWARE em algumas aplicações desenvolvidas por nosso grupo de trabalho como [Irene Ginani and da Silva 2017] e [Cavalcante and da Silva 2017], e por outros membros do projeto, com foco particular nos componentes de segurança: o *Identity Management* GE, que tem como implementação de referência o Keyrock, é responsável por aspectos

[1]https://www.fiware.org/
[2]https://smartmetropolis.imd.ufrn.br

de autenticação, enquanto que o *Authorization PDP* GE e *PEP Proxy* GE tratam aspectos de autorização. Além da autenticação o Keyrock é responsável pela gerência de usuários e de políticas de controle de acesso, permitindo a definição de papéis e permissões.

Com a experiência no uso destes componentes, identificamos algumas dificuldades na utilização do Keyrock, como por exemplo em operações de gerenciamento de usuários, e definição de políticas de controle de acesso com um número elevado de permissões e papéis. Partindo desse pressuposto, esse trabalho apresenta uma proposta de interface gráfica para gerenciamento de usuários e de políticas de controle de acesso baseado no Keyrock.

2. A Plataforma FIWARE

A plataforma FIWARE é vista como um *middleware*, que dispõe de diversos componentes reutilizáveis (denominados habilitadores genéricos ou *generic enablers* - GEs) para facilitar o desenvolvimento de aplicações [Di Cola et al. 2015]. Os componentes FIWARE são agrupados em capítulos técnicos[3] em torno de temas específicos, como IoT ou segurança. Sendo nosso foco os habilitadores genéricos referentes a segurança, mais especificamente, o *Identity Management* GE, o *Authorization PDP* GE e o *PEP Proxy* GE.

O Identity Management (Idm) GE é um componente de autenticação, que provê serviços para as aplicações, fazendo a administração de usuários, papéis e permissões. O Keyrock possui um backend desenvolvido em Node.js, e queu oferece uma API [FIWARE 2018] a qual permite que os desenvolvedores possam implementar novas funcionalidades ou adaptá-las para algumas aplicações mais específicas. Por outro lado, o AuthZForce tem como principal objetivo armazenar e tomar decisões sobre as políticas de segurança criadas pelo usuário no Keyrock. Assim os papéis, permissões e as atribuições de papéis a usuários que são cadastradas na interface são traduzidos para XACML onde o authzforce interpreta a política. Quando ocorre alguma requisição a um recurso na aplicação, que utiliza esse componente, a requisição é enviada para que a GE possa calcular se aquele usuário tem acesso aquele recurso em questão [Edition 2017].

Em alternativa ao Keyrock, temos o Horizon da plataforma Openstack[4], que unido ao keystone realiza as operações de administração de usuários além de ter uma API com mais funcionalidades, facilitando o desenvolvimento de ações que a utilizam. Entretanto, a interface gráfica do Horizon também apresenta funcionalidades limitadas, não atendendo às demandas que motivaram a realização deste trabalho.

Ao utilizar esses componentes para proteger aplicações percebemos algumas falhas, como a falta de mecanismos de gerenciamento de usuários (como edição e deleção), e habilitação de usuários após seu cadastro. Outro problema detectado foi a limitação na construção de políticas, de forma que não é possível ocorrer uma herança de papéis, ocasionando uma possível explosão de papéis, tendo como proposta implementar, futuramente, o RBAC hierárquico que solucionaria esse problema. Com isso, identificamos uma limitação no uso destes componentes por parte de desenvolvedores, o que nos levou a buscar maneiras de simplificar a utilização dos componentes de segurança da plataforma FIWARE.

[3] `https://catalogue-server.fiware.org/`
[4] `https://openstack.org`

3. Solução Proposta

Assim para que possamos solucionar os problemas encontrados nossa proposta foi de implementar uma interface Web para auxiliar na utilização do keyrock. Com isso, buscamos permitir a desenvolvedores facilidades para a administração de usuários e outras aplicações, e para a definição de políticas de controle de acesso.

3.1. Arquitetura da Solução

Para entender melhor o funcionamento da aplicação proposta observemos sua arquitetura na Figura 1. Na arquitetura proposta, temos o seu backend feito em Python, que abstrai e simplifica o consumo da API do keyrock, e um frontend Web que utiliza nosso backend para se comunicar com o keyrock. Com isso, temos também uma biblioteca Python que pode ser utilizada por desenvolvedores que desejem interagir com a API do keyrock.

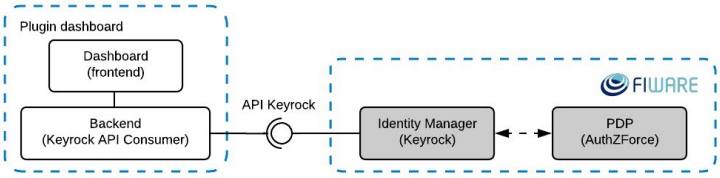

Figura 1. Arquitetura da aplicação proposta

Além das funcionalidades implementadas utilizando a API, também garantimos que o usuário esteja autenticado para acessar as informações dos usuários, que são acessadas quando a autenticação é realizada. Assim, o *token* do usuário é verificado para certificar que este pode ter acesso aos dados que serão exibidos.

3.2. Detalhes da Implementação

A interface Web foi desenvolvida com Python utilizando-se do framework django, consumindo nosso backend python. Também foi utilizado docker para criar um ambiente virtual para a aplicação. Utilizando html, css, javascript e bootstrap foi desenvolvido o frontend da aplicação, uma página simples foi feita para tela de início, e após a autenticação teremos a principal tela do sistema como visto na Figura 2.

Figura 2. Exibição dos Usuários

4. Conclusão

No decorrer do desenvolvimento, conseguimos entregar uma versão funcional que está em uso pelo projeto SmartMetropolis, suprindo algumas das necessidades já destacadas anteriormente. Mas apesar do seu uso e de suprir algumas necessidades do Keyrock, ainda são necessárias que algumas das melhorias sejam implementadas.

Assim, como trabalho futuro, pretendemos incrementar a interface para que ela possa suprir todas as necessidades encontradas no Keyrock, em relação aos papéis e permissões, como a necessidade do RBAC hierárquico, fase esta que está em andamento para que a GUI possa ser mais completa ao suprir as necessidades do Keyrock.

Referências

Andriani, P., Briguglio, L., Lombardo, L., Nigrelli, M., Pellegrino, D., Torres, J. S., and Voulkidis, A. (2015). Fiware generic enablers as building blocks of a marketplace for energy. In *eChallenges e-2015 Conference*, pages 1–10.

Cavalcante, G. and da Silva, C. E. (2017). Uma proposta de arquitetura para autorização federada com internet das coisas. SBSEG 2017 - Simpósio Brasileiro de Segurança da Informação e Sistemas Computationais.

Di Cola, S., Tran, C., Lau, K.-K., Celesti, A., and Fazio, M. (2015). A heterogeneous approach for developing applications with fiware ges. In Dustdar, S., Leymann, F., and Villari, M., editors, *Service Oriented and Cloud Computing. ESOCC 2015.*, number 9306 in Lecture Notes in Computer Science, pages 65–79, Cham. Springer International Publishing.

Edition, A. C. (2017). Authzforce fiware documentation. `https://authzforce-ce-fiware.readthedocs.io/en/latest/`.

FIWARE (2018). Api do keyrock. `https://keyrock.docs.apiary.io/#reference`.

García Vázquez, A., Soria-Rodriguez, P., Bisson, P., Gidoin, D., Trabelsi, S., and Serme, G. (2011). Fi-ware security: Future internet security core. In Abramowicz, W., Llorente, I. M., Surridge, M., Zisman, A., and Vayssière, J., editors, *ServiceWave Towards a Service-Based Internet*, volume 6994 of *Lecture Notes in Computer Science*, pages 144–152. Springer Berlin Heidelberg.

Irene Ginani, G. C. and da Silva, C. E. (2017). Proposta de interface gráfica web para gerenciamento de usuários na plataforma fiware. X EPOCA - Escola Potiguar de Computação e suas Aplicações.

Moltchanov, B. and Rocha, O. R. (2014). Generic enablers concept and two implementations for european future internet test-bed. In *2014 International Conference on Computing, Management and Telecommunications (ComManTel)*, pages 304–308.

Sandhu, R. S. and Samarati, P. (1994). Access control: principle and practice. *IEEE Communications Magazine*, 32(9):40–48.