Implementação dos mecanismos de gerenciamento de identidades, autenticação e autorização no Helix Sandbox NG

Felipe Matheus da Cunha [1]
João Vitor de Araújo Bonfim [2]
Matheus Pereira Mendes [3]
Fábio Henrique Cabrini [4]

Resumo:

Este trabalho tem o objetivo de implementar a arquitetura de segurança proposta pela FIWARE no Helix Sandbox NG, os Generic Enables são Keyrock, Wilma PEP-Proxy e AuthzForce que oferecem os serviços de gerenciamento de identidades, autenticação e autorização respectivamente. Também foi criado um repositório no GitHub que inclui os arquivos necessários para implementar a arquitetura de segurança juntamente com o tutorial completo para o processo de instalação e configuração das ferramentas.

Palavras chave: Wilma PEP Proxy, Keyrock, AuthzForce.

Abstract:

This work aims to implement the security architecture proposed by FIWARE in Helix Sandbox NG, generic enables are Keyrock, Wilma PEP-Proxy and AuthzForce that offer identity management, authentication and authorization services respectively. A repository on GitHub has also been created that includes the files needed to implement the security architecture along with the full tutorial for the tool installation and configuration process.

Keywords: Wilma PEP Proxy, Keyrock, AuthzForce.

Introdução

Com o avanço na tecnologia de comunicação 5G, os dispositivos inteligentes estão ganhando cada vez mais destaque, principalmente com o conceito de cidades inteligentes, onde hoje é possível espalhar sensores por cidades inteiras e realizar o monitoramento da cidade, como por exemplo, barulho, trânsito, estruturas de pontes etc.

A segurança nessas plataformas vem se fazendo cada vez mais necessária para garantir a privacidade, integridade e disponibilidade, pois muitas vezes são geradas informações sensíveis.

Existem diversas plataformas para o desenvolvimento de ambientes inteligentes, como por exemplo a AWS (Amazon Web Services) IoT (Internet of Things) desenvolvida pelo Amazon, Google Cloud, entre outras empresas que desenvolvem soluções para ambientes IoT. Um exemplo de empresa que desenvolve soluções de IoT para parques de rede é o caso da Cisco System que produz

[1] Graduado em Segurança da Informação. E-mail: felipe.cunha2@fatec.sp.gov.br
[2] Graduado em Segurança da Informação. E-mail: joao.bonfim2@fatec.sp.gov.br
[3] Graduado em Segurança da Informação. E-mail: matheus.mendes12@fatec.sp.gov.br
[4] Doutor e Mestre em Engenharia Elétrica. E-mail: fabio.cabrini@fatec.sp.gov.br

equipamentos com funções nativas para dispositivos IoT, como por exemplo os Switches ISR (Integrated Services Router) focados em ambientes industriais.

Existem projetos focados em desenvolvimento de ambientes IoT como é o caso da FIWARE *Foudantion* que é um projeto financiado pela União Europeia que desenvolve as soluções que serão utilizadas no desenvolvimento desse trabalho. Para ambientes inteligentes com base em IoT, são desenvolvidas plataformas que buscam melhorar o desempenho da comunicação entre os dispositivos e a aplicação que receberá os dados, como por exemplo a plataforma Helix Sandbox NG, que segundo Cabrini (CABRINI, FILHO, et al., 2019) é uma plataforma que orquestra os principais GEs (Generic Enablers) desenvolvidos e mantidos pela FIWARE Foundation.

Visando ser prática, a plataforma Helix Sandbox NG é utilizada em ambientes de desenvolvimento de aplicações para ambientes inteligentes e IoT, por se tratar de uma plataforma focada na prototipação rápida, sendo indicada para o desenvolvimento de PoCs (Proof of Concept), MVPs (Minimum Viable Product) e pesquisas acadêmicas, não possui camadas de segurança, com isso foi aplicado métodos de segurança eficazes, que garantam a integridade por meio de sistemas de autenticação.

Por se tratar de uma plataforma para prototipagem, existem diversas vulnerabilidades para serem exploradas, dedicada ao estudo de soluções IoT, se torna a base de diversos projetos e trabalhos, que são desenvolvidos em instituições de ensino e pesquisa.

Internet das Coisas

Com o foco de compreender os fenômenos que ocorrem no mundo ao nosso redor, tecnologias vêm sendo desenvolvidas com a finalidade de facilitar a vida cotidiana. Devido ao aumento de dispositivos inteligentes que necessitam de meios de comunicação para enviar os dados por eles obtidos, as tecnologias de comunicação vêm avançando rapidamente com o passar do tempo. Com este aumento de dispositivos, ocorre uma demanda maior de conexões a internet, principalmente pelo conceito de cloud. Com a ocorrência deste fenômeno e necessidade do mundo moderno, o termo Internet das Coisas vem ganhando relevância. (GALEGALE, SIQUEIRA, *et al.*, 2016).

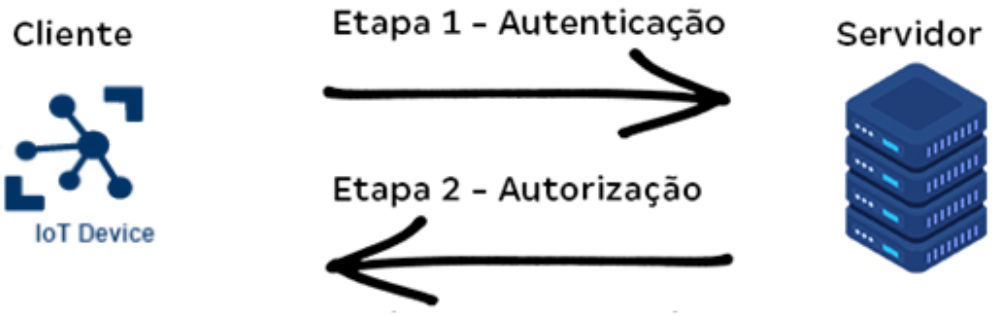
Com o desenvolvimento de dispositivos IoT se tornando foco de novos projetos em todos os setores, a segurança se torna cada vez mais necessária, pois diversos dados são gerados a todo momento, e muitos são críticos e confidenciais.

Autenticação, Autorização e Contabilização

O AAA (*Authentication, Authorization, and Accounting*) refere-se a um conjunto de mecanismos de segurança que se baseia em autenticação, autorização e contabilização. Autenticação diz respeito a um método que será utilizado para identificar qual usuário está tentando se autenticar através de *tokens* podendo o acesso ser permitido ou negado. Autorização refere-se ao que os usuários podem ter acesso, e por último a contabilização que diz respeito as contas que serão utilizadas pelos usuários para efetuar *logon* no sistema (TARGET, 2010).

Como demonstrado na Figura 1, o sistema consiste em duas etapas, sendo a primeira o usuário informar os seus dados de acesso, a partir dessa informação o sistema irá consultar a base de dados para garantir que o usuário de fato existe e pode acessar, com essa autorização obtida, o sistema passa para a segunda etapa do processo, que consiste na leitura de regras criadas previamente, que irão informar se o usuário possui autorização para fazer as ações no ambiente.

Figura 1 - Autenticação e autorização



Fonte: Autoria própria, 2021.

Com esse sistema, é possível criar regras para diversas finalidades, como acesso ao servidor de arquivos, e até mesmo criar regras para acesso ao Wi-Fi de uma rede, determinando quem pode acessar.

Token

Os *tokens* são senhas únicas que identificam o dispositivo, onde com base nesse código único que o dispositivo recebe, ele será identificado na rede, em alguns casos os *tokens* são senhas que expiram com o tempo pré-determinado pelo dispositivo gerador de *token* que o usuário está utilizando, os *tokens* ou aplicativos que são muito utilizados quando é ativado o MFA (*Multi-Factor Authentication)*.

XACML

XACML *(eXtensible Access Control Markup Language)* é uma linguagem de política de controle de acesso sendo uma OASIS (*Organization for the Advancement of Structured Information Standards*) *standard,* possuindo um foco tanto de políticas como controle de acesso, visando facilitar as questões de segurança em códigos, o XACML permite gerenciar os requerimentos de controle de acesso (HAL LOCKHART, 2020), onde a requisição/resposta, irá sofrer a ação de ser autorizada ou negada, para se obter esse resultado, a linguagem é baseada em 4 valores, *Permit,* que libera o acesso do indivíduo ao ambiente, *Deny,* que irá negar o individual impossibilitando assim que ele realize ações no ambiente, *Inderterminate,* sendo esse um erro ou quando algum requerimento está faltando, e por fim o valor *Not Applicable*, onde a requisição não será respondida pelo serviço.

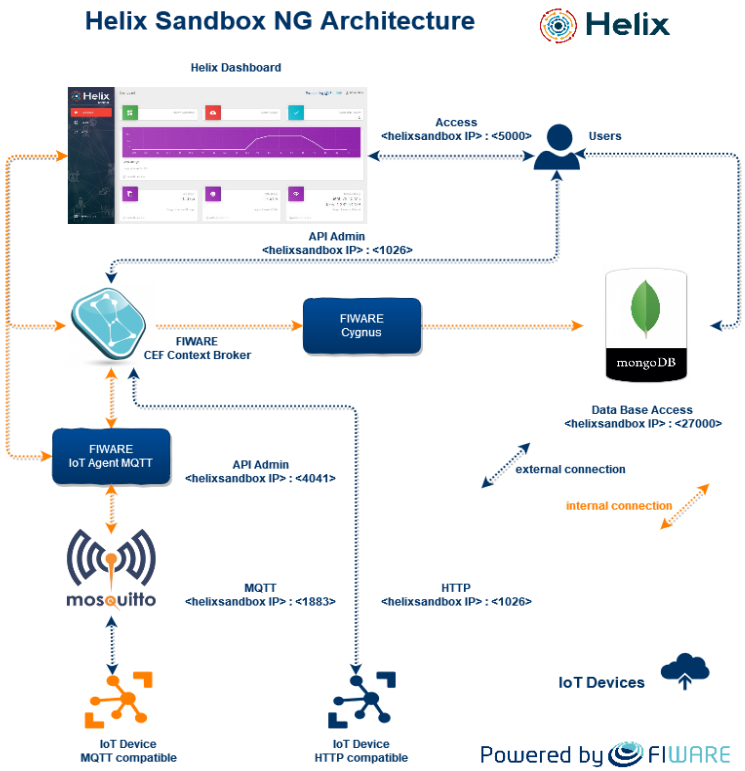
Helix Sandbox NG

A plataforma de *back-end* Helix Sandbox NG que de acordo com (CABRINI, FILHO, *et al.*, 2019), foi desenvolvida para realizar a prototipação de aplicações IoT, utilizando como base alguns módulos, também conhecidos como *Generic Enablers* (GEs) do FIWARE *e mantidos pela* FIWARE *Foundation*. A plataforma Helix foi certificada em 2018*,* e recebeu o certificado *Powered By* FIWARE*.* Por se tratar de uma plataforma *sandbox*, focada em desenvolvimento de novas soluções IoT*,* foi criada com objetivo de reduzir a curva de aprendizagem, utilizando poucos recursos computacionais, orquestração e configuração automáticas dos GEs e possuir uma interface gráfica de gerenciamento intuitiva.

Sendo uma plataforma para o estudo e desenvolvimento de soluções IoT*,* possui uma estrutura de comunicação de fácil compreensão, suportando atualmente dois protocolos de comunicação, HTTP (*Hypertext Transfer Protocol*) e o MQTT (*Message Queue Telemetry Transport*).

Como demonstrado na Figura 2 a plataforma Helix Sandbox NG possui duas soluções da FIWARE integradas em sua arquitetura, que são o Orion Context Broker e o Cygnus.

Figura 2 - Fluxos de dados na Plataforma Helix Sandbox NG



Fonte: Cabrini, 2020.

O Orion é o principal componente da arquitetura FIWARE, sua função é realizar o gerenciamento e o processamento das informações de contexto, é um *broker* desenvolvido em linguagem C++, que implementa a API NGSIv2 REST, realizando o gerenciamento do ciclo de vida das informações produzidas pelos dispositivos de IoT, como atualizações, consultas, registros e assinaturas (ORION, 2020).

O *broker* torna as informações visíveis aos usuários, permitindo o seu gerenciamento, estabelecimento de *triggers* através de subscrições que ativam o mecanismo de notificação, recebendo assim informações quando determinados eventos pré-definidos estiverem ocorrendo.

O Cygnus é um conector que liga os dispositivos que geram dados aos bancos de dados, sendo assim possível criar um histórico de informações, permitindo a visualização das alterações realizadas no ambiente (CYGNUS, 2020).

Keyrock

É uma GE da FIWARE responsável pelo gerenciamento de identidades. Para realizar esse controle, as identidades são criadas dentro do Keyrock e armazenadas em um banco de dados, sendo utilizado nesse caso o MySQL, sempre que for necessário verificar uma identidade para autorizar

acesso no sistema, o Keyrock é o responsável por validar a identidade fornecida e verificar sua existência na base de dados, para um controle maior sobre as identidades, fornecendo assim mais segurança para o ambiente (KEYROCK, 2020).

É possível realizar a integração do Keyrock com mais duas soluções, também fornecidas pela FIWARE, que são: Wilma PEP Proxy e o AuthzForce, realizando essa integração das soluções, foi possível ativar o protocolo OAuth2.

Wilma PEP Proxy

É um GE que exerce a função de PEP Proxy, sendo esse responsável por gerenciar a comunicação entre os dispositivos end-to-end para aumentar a segurança do ambiente, é possível realizar a integração com outras duas soluções da FIWARE (WILMA, 2020), o Keyrock e AuthzForce, com essas soluções integradas, toda a comunicação realizada no ambiente irá passar por ele, sendo responsável pelo gerenciamento dessa comunicação, onde para liberar ou não o acesso do dispositivo à informação, será realizada consultas tanto no Keyrock para validar a identidade do dispositivo, como no AuthzForce, para verificar as permissões de acesso do dispositivo no ambiente.

AuthzForce

O AuthzForce é uma API (Application Progamming Interface) desenvolvida pela FIWARE Foundation, cujo sua função é o controle de autorização para ações, em que com sua utilização será possível gerenciar o que os indivíduos podem fazer no ambiente, se baseando em políticas de acesso pré-criadas (AUTHZFORCE, 2020).

Para seu funcionamento o AuthzForce é baseado na arquitetura REST (Representational State Transfer), sendo essa uma arquitetura de comunicação entre cliente e servidor, onde é estabelecida uma comunicação TCP/IP e são realizadas requisições GET, sendo essa muito comum em aplicações web, para compilar as informações geradas o AuthzForce utiliza o XACML v3.0.

Vulnerabilidades da Plataforma Helix Sandbox NG

A plataforma Helix possuí algumas vulnerabilidades na sua estrutura, por se tratar de uma plataforma para o desenvolvimento de projetos IoT, não possuí mecanismos de autenticação, ou seja, qualquer pessoa pode realizar requisições para a plataforma se souber o IP e alterar os dados dos dispositivos IoT válidos.

Para corrigir essa vulnerabilidade, foi utilizado as soluções da FIWARE Keyrock, Wilma PEP Proxy e AuthzForce, que quando integradas a plataforma estabelecerão um sistema de autenticação,

onde somente pessoas e dispositivos autorizados poderão realizar consultas a base de dados e realizar alterações. Para controlar as ações no ambiente foi utilizado o AuthzForce, que acrescentou a camada de autorização no ambiente, onde as ações passaram a ser controladas por regras previamente estabelecidas.

Uma outra vulnerabilidade que não foi tratada é a falta da presença de um firewall, pois ainda é possível se comunicar diretamente com a plataforma Helix Sandbox sem a necessidade de passar pelo Wilma, através de realizações de requisições na porta 1026.

Integração das soluções na plataforma Helix Sandbox NG

Como início do desenvolvimento do projeto, o primeiro passo consiste na compreensão da plataforma Helix Sandbox NG, juntamente com a função do Orion Context Broker, responsável pelo gerenciamento de informações de contexto gerados pelos dispositivos no ambiente. Com esses objetivos definidos o segundo passo foi a instalação da plataforma em um ambiente de VMs (*Virtual Machine*), seguido da realização de testes focando na criação de entidades e alteração de suas informações.

Com esses testes foi possível compreender a responsabilidade da plataforma Helix Sandbox NG na prototipação do ambiente. A partir deste ponto ocorreu a compreensão do *context broker* no ambiente, sendo assim possível o entendimento do fluxo de comunicação entre o dispositivo IoT e Orion.

Para o início dos testes, foi criado um ambiente que consiste na utilização de VMs, utilizando a ferramenta Virtual Box, no caso foi criado apenas uma VM contendo 1GB de memória RAM e 15 GB de armazenamento, pois, como se trata de uma solução *lightweigh*t, não se fez necessário mais poder computacional ou divisão de trabalho. A instalação do Helix ocorreu conforme orientado pela própria documentação no GitHub (CABRINI, 2020) e de forma automatizada.

Durante os testes para a compreensão das soluções implementadas no ambiente ocorreu uma análise com o objetivo de verificar a autenticação e autorização necessária para realizar mudanças no banco de dados, onde foram feitas requisições pra criar entidades na base de dados, e também alterar os dados que elas possuem, sem informar no *Header* da requisição qualquer tipo de *token*, sendo constatado que não existe a necessidade do dispositivo se autenticar no ambiente para solicitar a leitura de informações no banco de dados ou publicar novas informações, ou mesmo para alterar informações de entidades já existentes no banco de dados.

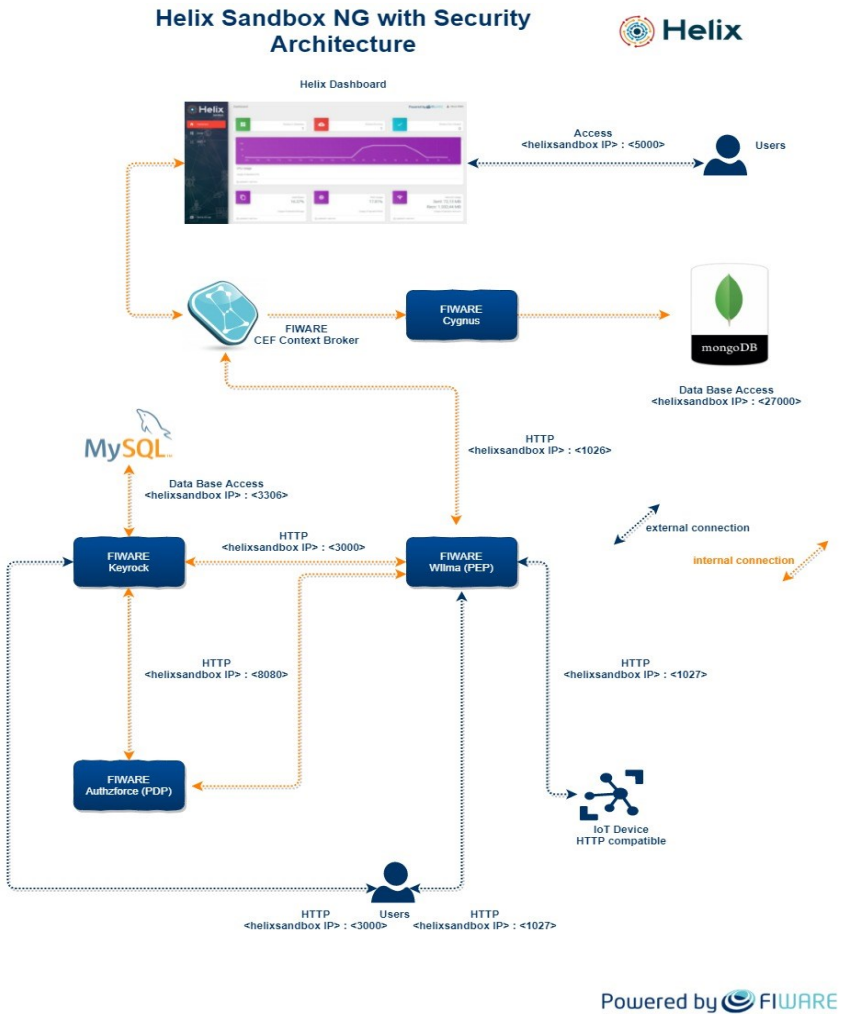
A partir dos resultados observados, ficou provado a necessidade da implementação de uma arquitetura de segurança que irá garantir a autenticação e autorização no ambiente, pois em nenhum momento foi solicitado qualquer meio de autenticação.

Com essa necessidade constatada foi dado início ao processo de implementação da arquitetura, partindo da compreensão das soluções que foram utilizadas.

Com a utilização da arquitetura proposta neste projeto, é possível realizar a instalação e configuração dos mecanismos de segurança, sem a necessidade de alterações no Orion Context Broker, pois o Wilma será responsável por receber a comunicação do dispositivo IoT, validar o acesso baseado nas identidades criadas pelo Keyrock, verificar as permissões de acesso no AuthzForce e repassar a comunicação para o Orion.

Na Figura 3 é possível visualizar a estrutura da arquitetura implementada, como suas conexões, os componentes e o fluxo de comunicação.

Figura 3 - Arquitetura de segurança integrada ao Helix Sandbox NG



Fonte: Autoria própria, 2021.

Como demonstrado na Figura 3, a arquitetura proposta no projeto foi implementada sem a necessidade de alterações na estrutura da plataforma Helix e no Orion Context Broker, onde o usuário passa a ter acesso a plataforma para visualização do Context Broker, porém quando for realizar a criação de entidades no Context Broker é necessário se autenticar no Wilma. Para realizar essa solicitação o usuário deve ser criado no Keyrock, sendo acessado através da porta 3000, e suas regras de acesso também são criadas e passados pelo usuário no Keyrock, sendo ele responsável por repassar as regras para o AuthzForce.

Anteriormente o dispositivo IoT quando se comunicava com a plataforma para realizar alterações, ocorria uma comunicação direta com o Context Broker sem a necessidade de autenticação. Com a implementação, é necessário que a comunicação passe pelo Wilma, onde o dispositivo IoT se comunica com o PEP Proxy através da porta 1027 informando seu token de autenticação. A partir deste ponto será feita uma consulta ao Keyrock para validar a existência do token e, após isso, confirmar as permissões de ações junto ao AuthzForce. As regras no AuthzForce juntamente com a entidade do dispositivo IoT, são criadas previamente pelo administrador do sistema por meio do dashboard de comunicação do Keyrock.

A instalação utilizou a tecnologia de *containers,* com isso se torna necessário alguns pré-requisitos que também devem ser instalados, o Docker e o Docker Compose*,* responsável pela criação e gerenciamento de *containers.*

O Keyrock pode ser instalado na forma de *container*, foi utilizado um arquivo do Docker-compose fornecido pela própria FIWARE na documentação (KEYROCK, 2020). Com a inicialização do *container,* foi carregado a imagem do Keyrock juntamente com o banco de dados que ele utiliza para armazenar as entidades.

Por meio de comandos fornecidos e explicados na documentação do Keyrock, foram realizados alguns testes criando entidades, tanto por linha de comando, como por meio de interface gráfica, com isso ocorreu a compreensão da utilização de *tokens* juntamente com o processo de criação.

No segundo momento, após a finalização dos testes e compreensão dos gerenciamentos de identidades por meio do Keyrock, foi iniciado o processo de instalação do Wilma PEP-Proxy, sendo ele responsável pela comunicação do Keyrock com todo o ambiente.

A instalação do Wilma também ocorreu a partir de *containers*, porém nesse caso não foi utilizado o método de Docker-compose, e sim a criação de uma imagem. Para realizar essa criação foi necessário utilizar o node.js, a partir de um Dockerfile fornecido pela FIWARE, foi criado a

imagem do *container* e com a utilização de um arquivo de configuração, também fornecido pela FIWARE, o *container* foi ativado, sendo ele configurado a partir do arquivo.

Para o Wilma algumas alterações são necessárias no arquivo de configuração, informar os dados de acesso do Keyrock, o IP, porta de acesso, juntamente com os *tokens* de acesso para se autenticar no ambiente.

Finalizado a configuração do arquivo, a partir da ferramenta Docker, ocorre a criação de um *containe*r baseado na imagem gerada anteriormente do Wilma e configurado a partir do arquivo .js, a constatação da instalação do Keyrock será feita através da análise de *logs* de conexão gerados tanto pelo Keyrock como pelo Wilma.

Constatado a conexão demonstrada na Figura 4, a próxima etapa consistiu na simulação de um dispositivo IoT tentando se comunicar com o context broker, onde foi testado se os dispositivos possuem a autorização para se comunicar no ambiente. Essa autorização foi testada por meio do Keyrock, que informa se o token está atrelado a uma identidade ou não.

Figura 4 - Log de conexão do Wilma

```
fiware-keyrock | Tue, 25 May 2021 13:55:14 GMT idm:api-pep_proxies --> authenticate
fiware-keyrock | Executing (default): SELECT `id`, `password`, `salt`, `oauth_clien
t_id` FROM `pep_proxy` AS `PepProxy` WHERE `PepProxy`.`id` = 'pep_proxy_b287f276-fa
82-42f6-af4d-19fea0cf7016';
fiware-keyrock | Executing (default): INSERT INTO `auth_token` (`access_token`,`exp
ires`,`valid`,`pep_proxy_id`) VALUES ('5213da4d-a3d0-4815-a1cd-6bdc25af3416','2021-
05-25 14:55:14',true,'pep_proxy_b287f276-fa82-42f6-af4d-19fea0cf7016');
fiware-keyrock | POST /v3/auth/tokens 201 96.205 ms - 138
```

Fonte: Autoria própria, 2021.

Ao realizar esse teste foi gerado um token, para gerar o token é necessário utilizar o ID e o secret da entidade, juntando ambos e criando um base 64 que consiste na concatenação dos dois tokens, assim convertendo para a base 65. Com o base 64 gerado, é realizado uma requisição ao Keyrock solicitando um token de acesso, onde a resposta obtida é o access token e o refresh token. Com isso foi possível realizar requisições ao context broker como demonstrado na Figura 5.

Figura 5 - Leitura da Plataforma Helix

```
fiware-keyrock | Executing (default): INSERT INTO `oauth_access_token` (`access_to
ken`,`expires`,`scope`,`valid`,`oauth_client_id`,`user_id`,`iot_id`,`refresh_token
`,`authorization_code`) VALUES ('2e1dd65c237175aa32aca64737cca551ad47be21','2021-0
5-25 15:03:58','bearer',true,'55a5b097-53ff-4afc-99ce-99d8a7c68164','admin',NULL,'
1ff96ad344143f45490421f853bb26084e873427',NULL);
fiware-keyrock | Executing (7f3a58b7-2f71-4ca9-ac9a-0078454f9e96): COMMIT;
fiware-keyrock | POST /oauth2/token 200 95.449 ms - 177
```

Fonte: Autoria própria, 2021.

Na terceira etapa ocorreu a instalação do AuthzForce, também utilizando da tecnologia de *container,* com a sua imagem sendo gerada a partir de um arquivo Docker-compose.yml.

O AuthzForce foi instalado por meio de *containers*, porém ele consome mais recursos das máquinas. Todas as soluções foram instaladas na mesma máquina, sendo necessário expandir a memória RAM (*Random Access Memory*) para 4GB. O arquivo de *container* também foi fornecido pela FIWARE, com a ativação do *container*, e nenhuma alteração é necessária no arquivo do AuthzForce.

A partir da ativação do AuthzForce foram necessárias alterações nos arquivos do Keyrock e do Wilma, no arquivo do Keyrock foi adicionado alguns argumentos que informam os dados de acesso do AuthzForce para o Keyrock, como o IP, porta, e a autenticação PDP (*Policy Decision Point*) foi configurada como avançada. No arquivo no Wilma a ativação é mais simples, necessário apenas informar que o responsável pelo PDP é o AuthzForce juntamente com o acesso, o IP e porta.

Os testes realizados para validar o funcionamento do AuthzForce consiste na criação de regras em XACML no Keyrock, sendo ele responsável por repassar as mesmas para o AuthzForce e gerar os domínios. A validação do funcionamento das soluções foi feita por meio da simulação de um dispositivo IoT tentando se comunicar com o context broker, e verificar se a autenticação foi consultada no Keyrock, sendo ela feita a partir da validação do token, e se as regras criadas no AuthzForce foram consultadas para garantir se o dispositivo possui permissão para realizar as ações no ambiente.

Para o primeiro teste foi realizado uma requisição sem a existência de uma regra, porém com o AuthzForce, e como constatado na Figura 6, retorna o erro que o domínio não existe, sendo assim a conexão é recusada.

Figura 6 – Conexão recusada

```
p://192.168.0.112:1027/v2/entities' \
> --header 'Accept: application/json' \
> --header 'fiware-service: helixiot' \
> --header 'fiware-servicepath: /' \
> --header 'X-Auth-Token: eb38ae036ea662997ddcb22f19e25bb0aac44e03'
AZF domain not created for application 7dabb0c8-7c5d-4192-84ae-7b9814a2c0fdroot@
root@helix:/home/helix/arquitetura-seguranca#
```

Fonte: Autoria própria, 2021

Para então confirmar que o AuthzForce controla os acessos as entidades contidas no *40ontexto broker*, foi realizado um segundo teste, onde foi criada uma regra no Keyrock que permitia ao dispositivo realizar as leituras, e como constado na Figura 7, após a criação da regra, a mesma solicitação foi realizada, obtendo as informações da entidade.

Figura 7 – Conexão aprovada

```
":["bearer"]}root@helix:/home/helix/arquitetura-seguranca/wilma# curl --location -
'http://192.168.0.112:1027/v2/entities' \
> --header 'Accept: application/json' \
> --header 'fiware-service: helixiot' \
> --header 'fiware-servicepath: /' \
> --header 'X-Auth-Token: b97ad9629436445c5372de4d99d74c1bce522a70'
[{"id":"urn:ngsi-ld:entity:001","type":"iot","humidity":{"type":"float","value":0,
"metadata":{}},"temperature":{"type":"float","value":0,"metadata":{}}}]root@helix:
/home/helix/arquitetura-seguranca/wilma#
```

Fonte: Autoria própria, 2021

Com o ambiente implementado e validado, com o intuito de facilitar e auxiliar futuros projetos que irão utilizar a plataforma, foi gerado um GitHub contendo os arquivos necessários para instalação e configuração das soluções, juntamente com um tutorial escrito contendo um passo a passo para a instalação. Foi gerado um repositório no GitHub em português (https://github.com/felipe-mcunha/arquitetura-seguranca) e em inglês (https://github.com/m-mendes/security-architecture).

Resultados obtidos

Após a implementação dos métodos de autenticação, a partir dos testes realizados, constatou-se que as soluções estavam funcionando corretamente de maneira integrada, sendo somente possível realizar alterações no Helix, comunicando-se através do Wilma PEP Proxy informando o *access token*.

As requisições a Plataforma Helix Sandbox NG devem passar pelo Wilma, onde o dispositivo IoT tem a necessidade de informar um *token* para garantir a sua autenticação no ambiente, pois o Keyrock será consultado para garantir a veracidade do *token*.

O dispositivo só passou a realizar ações no ambiente após a criação de uma regra no Authzforce, mostrando assim um controle na autorização de ações dentro do ambiente, elevando o nível de segurança.

Considerações finais

A arquitetura desenvolvida para esse projeto, demonstrou-se eficiente e cumprindo os objetivos propostos para o projeto, onde a partir da sua instalação não será mais possível que pessoas alterem informações na Plataforma Helix, tornando também a instalação mais simples, sem a necessidade de realizar configurações na Plataforma Helix.

Com o objetivo de facilitar e centralizar a instalação de soluções de segurança da FIWARE, o desenvolvimento de um repositório no GitHub atingiu um nível satisfatório, onde se faz possível

encontrar os arquivos necessários e o passo a passo da instalação, assim auxiliando no desenvolvimento de projetos futuros.

Além de servir como um referencial para que os usuários da plataforma possam implementar os recursos de segurança previstos pela FIWARE *Foundation*.

Referências

A Brief Introduction to XACML. OASIS - Organization for the Advancement of Structured Information Standards, 2003. Disponível em: https://www.oasis-open.org/committees/download.php/2713/Brief_Introduction_to_XACML.html. Acesso em: 5 nov. 2020.

AUTHZFORCE'S OFFICIAL DOCUMENTATION. FIWARE. Disponível em: https://authzforce-ce-fiware.readthedocs.io/en/latest/. Acesso em: 21 abr. 2021.

CABRINI, F. H. **Helix Sandbox Next Generation**. GitHub, 2020. Disponível em: https://github.com/Helix-Platform/Sandbox-NG. Acesso em: 25 abr. 2021.

CABRINI, F. H. et al. **Helix SandBox: An Open Platform to Fast Prototype Smart Environments Applications**. IEEE, Arequipa, Peru, Peru, 26-29 Agosto 2019.

CABRINI, F. H. et al. **SMART BABY: APLICAÇÃO DOS CONCEITOS DA INTERNET DAS COISAS (IoT) PARA PREVENÇÃO DE ACIDENTES NA INFÂNCIA**. FTT Journal of Engineering and Business, São Bernardo do Campo, v. 5, n. 5, p. 62-74, Dezembro 2019. ISSN 2525-8729.

CAIAFA, J. **Automação e agência humana na Linha 4-Amarela do metrô de São Paulo**. Galáxia, São Paulo, n. 29, p. 83-95, Junho 2015.

CONTEXT BROKER. **European Union**. Disponível em: https://ec.europa.eu/cefdigital/wiki/display/CEFDIGITAL/Context+Broker. Acesso em: 7 nov. 2020

FIWARE ORION. **FIWARE**. Disponível em: https://fiware-orion.readthedocs.io/en/master/. Acesso em: 21 abr. 2021.

HAL LOCKHART, C. BILL PARDUCCI, C. **OASIS eXtensible Access Control Markup Language (XACML) TC. OASIS OPEN**. Disponível em: https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=xacml. Acesso em: 20 abr. 2021.

PUSTOKHINA, I. V. et al. **An Effective Training Scheme for Deep Neural Network in Edge Computing Enabled Internet of Medical Things (IoMT) Systems**. IEEE Access, v. 8, p. 107112–107123, 2020. Acesso em 5 jun. 2020.

IDENTITY MANAGER - KEYROCK. **FIWARE**. Disponível em: https://fiware-idm.readthedocs.io/en/latest/. Acesso em: 7 nov. 2020.

PEP PROXY - WILMA. **FIWARE**. Disponível em: https://fiware-pep-proxy.readthedocs.io/en/latest/. Acesso em: 7 nov. 2020.

TARGET, T. **Authentication, authorization, and accounting (AAA)**. TechTarget, 2010. Disponível em: https://searchsecurity.techtarget.com/definition/authentication-authorization-and-accounting. Acesso em: 15 mai. 2021.