Interoperabilidade e Segurança na Implementação de Aplicações Web de Saúde com SMART on FHIR

Interoperability and Security in the Implementation of Health Web Applications with SMART on FHIR

Interoperabilidad y Seguridad en la Implementación de Aplicaciones Web de Salud con SMART en FHIR

Leonardo Souza dos Santos[1], Gabriela del Mestre Martins[2], Fábio Pires Itturriet[1], Juliano Costa Machado[1], André Luís del Mestre Martins[1]

1 IFSul campus Charqueadas, Charqueadas (RS), Brasil.
2 Unimed Litoral, Balneário Camboriú (SC), Brasil.

Autor correspondente: André Luís del Mestre Martins
E-mail: andre.martins@ifsul.edu.br

Resumo

Objetivo: Desenvolver um servidor em nuvem para possibilitar a interoperabilidade entre diferentes aplicações web em saúde, com foco em um protocolo de autenticação para aplicações *IoT* (*Internet of Things*) e aplicações com interface de usuário. **Método:** O servidor envia e recebe dados no padrão FHIR (*Fast Healthcare Interoperability Resources*). O servidor autentica aplicações com interface de usuário utilizando *SMART (Substitutable Medical Apps and Reusable Technologies) on FHIR* enquanto aplicações *IoT* utilizam um cadastro prévio para autenticar com *SMART on FHIR*. **Resultados:** Uma aplicação FHIR desenvolvida por terceiros foi modificada para realizar a autenticação utilizando o servidor proposto ao invés do servidor original. Um protótipo de dispositivo IoT foi desenvolvido para realizar autenticação com o servidor. O servidor realizou com sucesso a autenticação de ambas aplicações. **Conclusão:** Aplicação de terceiros e dispositivos *IoT* se comunicando com o servidor *SMART on FHIR* desenvolvido comprovam a interoperabilidade e a segurança deste trabalho.

Descritores: Interoperabilidade; Privacidade dos Dados do Paciente; Saúde Digital

Abstract

Objectives: Develop a cloud server to enable interoperability between different healthcare web applications regarding authentication protocol for IoT (Internet of Things) applications and user interface applications. **Method:** TThe server sends and receives data in the FHIR (Fast Healthcare Interoperability Resources) standard. The server authenticates applications with user interfaces using SMART (Substitutable Medical Apps and Reusable Technologies) on FHIR while IoT applications are registered in advance to authenticate with SMART on FHIR. **Results:** A third-party FHIR application was modified to authenticate by using the proposed server instead of the original one. A prototype IoT device was developed to perform authentication with the server. The server successfully authenticated both applications. **Conclusions:** Third-party applications and IoT devices communicating with the proposed SMART on FHIR server prove the interoperability and security of this work.

Keywords: Interoperability, Confidentiality, Telemedicine

Resumen

Objetivo: Desarrollar un servidor en la nube para permitir la interoperabilidad entre diferentes aplicaciones web de atención médica, centrándose en un protocolo de autenticación para aplicaciones *IoT* (*Internet of Things*) y aplicaciones de interfaz de usuario. **Método:** El servidor envía y recibe datos en el estándar FHIR (*Fast Healthcare Interoperability Resources*). El servidor autentica aplicaciones con interfaz de usuario utilizando *SMART (Substitutable Medical Apps and Reusable Technologies) on FHIR*, mientras que las aplicaciones *IoT* utilizan un registro previo para autenticarse con SMART en FHIR. **Resultados:** Se modificó una aplicación FHIR desarrollada por un tercero para realizar la autenticación utilizando el servidor propuesto en lugar del servidor original. Se desarrolló un prototipo de dispositivo *IoT* para realizar la autenticación con el servidor. El servidor autenticó con éxito ambas aplicaciones. **Conclusión:** la aplicación de terceros y los dispositivos *IoT* que se comunican con el servidor *SMART on FHIR* desarrollado demuestran la interoperabilidad y la seguridad de este trabajo.

Descriptores: Interoperabilidad; Confidencialidad; Telemedicina

Introdução

A interoperabilidade entre sistemas de informação em saúde é um dos principais desafios para utilização plena dos Prontuários Eletrônicos de Paciente (PEPs). Dentre os diversos padrões para troca de dados já propostos visando a questão da interoperabilidade, *Fast Healthcare Interoperability Resources*[1] (FHIR) se destaca pela crescente adoção mundial, incluindo a Rede Nacional de Dados em Saúde (RNDS) no Brasil.

FHIR possui o formato de envio de dados estruturado nos chamados Recursos (*Resources*) e é desenvolvido para ser integrado por aplicações web com modelo de arquitetura REST (*Representational State Transfer*), que se alinha fortemente com as atuais práticas tecnológicas empregadas no desenvolvimento de software para web[2]. Uma limitação do FHIR é a ausência de definição sobre o controle de acesso e a segurança dos PEPs. Fragilidade ainda mais relevante quando considerada as leis de proteção de dados (LGPD no Brasil e GDPR na Europa). Parte das informações dos PEPs são confidenciais e simplesmente não podem ser violadas, logo não devem estar transitando pela internet sem qualquer tipo de proteção[3].

Diferentes soluções para prover a troca de dados e PEPs de forma segura são encontradas na literatura[4-6], mas a organização HL7 recomenda o SMART (*Substitutable Medical Applications and Reusable Technologies*), um perfil OAuth2 utilizado para consumir os Recursos de um servidor FHIR[4]. Em síntese, o *SMART on FHIR* estabelece um método para aplicações web de saúde com interface com usuário se conectarem a servidores FHIR na internet para a troca de Recursos FHIR com a segurança adequada[7].

Além das aplicações com interface com usuários, o ecossistema de sistemas de informação em saúde estão cada vez mais heterogêneos, integrando dispositivos *Internet of Things* (*IoT* – Internet das Coisas), inteligência artificial, computação de borda, nuvens públicas e privadas[8]. O desenvolvimento de servidores seguros e interoperáveis para atender aplicações web de saúde é bastante desafiador e fundamental para a viabilização plena da informática em saúde[9-10]. A elevada heterogeneidade desses sistemas introduz mais um grau de complexidade considerando a interoperabilidade e a segurança. Embora o *SMART on FHIR* seja projetado especificamente para oferecer suporte a

desenvolvedores que estão criando aplicativos com interface de usuário, ele não cobre todo o ecossistema de aplicações web de saúde, deixando de fora principalmente as aplicações *IoT*.

O objetivo deste trabalho é desenvolver, testar e disponibilizar *H₂Cloud* (*Heterogeneous Health Cloud* – Nuvem de Saúde Heterogênea), um servidor na nuvem para autenticação de aplicações web heterogêneas de saúde, compatível com *SMART on FHIR*.

Existem várias aplicações *SMART on FHIR* para diferentes propósitos[11-14] e, em comum nos trabalhos relacionados, é a existência de um servidor dedicado para uma aplicação em particular. Não é possível determinar se os servidores utilizados são úteis em outras aplicações[11]. Além disso, os trabalhos se concentram majoritariamente em aplicações com interface de usuário, ignorando parte do ecossistema de saúde[12-14].

O grande diferencial deste trabalho é utilizar *H₂Cloud* como servidor de aplicações de terceiros e dispositivos *IoT* para demonstrar a interoperabilidade para sistemas de informática em saúde. Favorável aos objetivos expostos no documento de Estratégias de Saúde Digital para o Brasil[15] (ESDB), as principais contribuições deste trabalho são:

a. A implementação prática de *H₂Cloud* integrado a um aplicativo de terceiro[1] que permite a troca de PEPs no padrão FHIR de forma segura;
b. A proposta de implementação do protocolo de autenticação SMART on FHIR em aplicações IoT;
c. A disponibilização do código-fonte de *H₂Cloud* de forma aberta e gratuita.

Métodos

A Figura 1 ilustra a visão geral de *H₂Cloud* e os variados tipos de aplicações web de saúde previstas na comunicação. A direita da Figura 1, aplicações com interface de usuário (sites, aplicativos, etc…) tem *endpoints* específicos para realizar a autenticação com *SMART on FHIR*. Uma vez autenticadas, as aplicações *SMART on FHIR* trocam dados com *H₂Cloud* usando recursos FHIR. A Esquerda da Figura 1, aplicações *IoT* também trocam dados com *H₂Cloud* utilizando recursos FHIR, mas tem um protocolo de

[1] https://apps.smarthealthit.org/apps/featured

autenticação específico, pois não é possível utilizar *SMART on FHIR* para esses casos. O processamento e o armazenamento dos recursos FHIR habilitam a troca de dados entre as diferentes aplicações e estão representadas no centro da nuvem na Figura 1. Os próximos parágrafos detalham cada um dos quatro blocos da nuvem *H₂Cloud.*

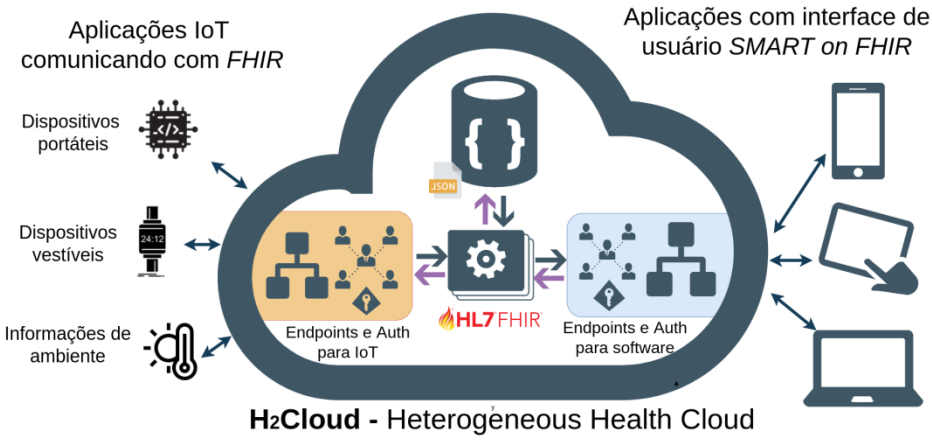

Figura 1 – Visão geral da arquitetura de software que o servidor em nuvem *H₂Cloud* visa atender.

Autenticação e *Endpoints* para Software com Interface de Usuário

Para implementar a autenticação de usuários no servidor *H₂Cloud*, o *SMART on FHIR* estabelece um fluxo de autenticação para que a aplicação web seja autorizada a acessar os Recursos FHIR que descrevem um PEP de um usuário específico.

O fluxo de autenticação em qualquer servidor que suporte *SMART on FHIR* é realizado por meio de requisições HTTP no padrão RESTful. Assim, *H₂Cloud* disponibiliza uma série de rotas HTTP que viabilizam a autenticação de qualquer aplicação compatível com *SMART on FHIR.* A Figura 2 apresenta o fluxo de autenticação entre aplicações *SMART on FHIR* e *H₂Cloud* e as principais requisições REST necessárias conforme segue:

1. **Informações para acesso do servidor.** Inicialmente, qualquer aplicação SMART deve procurar o arquivo *"wellknown"* do servidor, onde se encontram as informações para realizar todos os passos seguintes. Por exemplo, arquivo *"wellknown"* pode conter a especificação de como vai ocorrer a troca de *tokens,* as informações dos *endpoints* (Exemplo da Figura 2: `"token_endpoint" : "/auth/token"`) e os tipos de Recursos FHIR disponíveis.

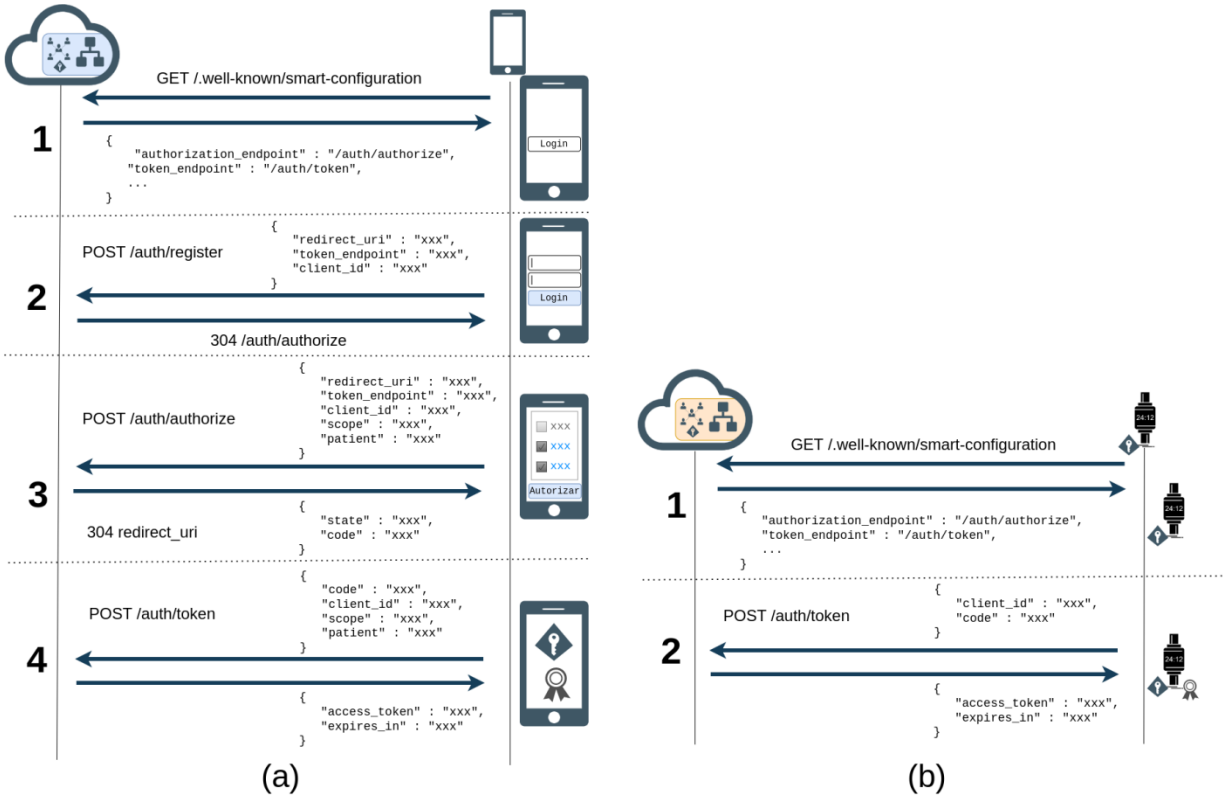

Figura 2 – Fluxo de autenticação *SMART on FHIR* implementado na nuvem *H₂Cloud* para (a) aplicações com interface de usuário e para (b) aplicações em dispositivos *IoT*.

2. **Registro da aplicação.** Após descobrir a URL responsável por iniciar o processo de autorização, a aplicação redireciona o usuário para essa URL passando os dados da aplicação para que *H₂Cloud* inicie o processo. Em seguida, *H₂Cloud*, ao receber essa solicitação, exibirá a tela de login, para que o usuário se identifique e possa reconhecer a aplicação solicitante. Ao verificar login e senha do usuário, *H₂Cloud* redireciona o usuário de volta para a URL de autorização.
3. **Autorização do usuário.** Quando o usuário tentar logar, *H₂Cloud* verifica se autoriza ou não o acesso do escopo de Recursos FHIR (`"scope"` na Figura 2). Os escopos ajudam a transmitir quais acessos um aplicativo precisa. Ao verificar o escopo, o servidor redireciona o usuário de volta para uma URL determinada pela aplicação (`redirect_uri` na Figura 2) junto de um um código de uso único (`"code"` na Figura 2).
4. **Geração de Token.** Após verificar login e senha, estar devidamente autorizado a acessar Recursos FHIR e adquirir código de uso único, a aplicação pode solicitar

um *token* para trocar Recursos FHIR com *H₂Cloud*. Finalmente, *H₂Cloud* retorna o *token* de acesso com validade específica ("`access_token`" e "`expires_in`" na Figura 2).

Autenticação e *Endpoints* para Dispositivos *IoT*

O *SMART on FHIR* estabelece um fluxo de autenticação para aplicações autônomas de terceiros pré-autorizadas onde a interação com o usuário não está prevista. O servidor *H₂Cloud* implementa uma variação desse fluxo para habilitar a autenticação de dispositivos *IoT.*

Assumindo que os dispositivos *IoT* não tem interface com usuários, eles devem ser previamente cadastrados e autorizados em *H₂Cloud* por um administrador. Além disso, o dispositivo *IoT* precisa desta mesma credencial própria, única e fixa salva em memória permanente. O processo de cadastro prévio do dispositivo *IoT* em *H₂Cloud* elimina os passos 2 e 3 previstos no fluxo completo SMART on FHIR (Figura 2).

Assim, o fluxo de autenticação para dispositivos *IoT* (Figura 3) fica reduzido a obtenção do *token* de acesso conforme segue:

1. **Informações para acesso do servidor.** O dispositivo *IoT* solicita o arquivo *"wellknown"* de *H₂Cloud*, onde encontra-se a especificação de como vai ocorrer a troca de *tokens e* as configurações de escopo dos Recursos FHIR disponíveis.
2. **Geração de Token.** Para gerar o token de acesso, o dispositivo envia suas credenciais prévias ("`client_id`" e "`code`" na Figura 3). Ao receber essa solicitação, *H₂Cloud* valida as credenciais e envia um *token* com a autorização prévia do dispositivo com validade específica.

Transmissão, Validação e Armazenamento dos Recursos FHIR

Ao receber o *token* devidamente assinado pelo servidor *H₂Cloud*, o processo de autenticação está concluído e a aplicação pode enviar e receber Recursos FHIR por meio de requisições REST. Para cada requisição realizada, *H₂Cloud* realiza um processo de avaliar a validade e a assinatura do *token* de acesso e verifica as permissões de acesso do usuário ou dispositivo *IoT*. Para identificar quais Recursos FHIR a aplicação possui acesso e qual as permissões de acesso (leitura e/ou escrita) do recurso solicitado, o

escopo autorizado pelo usuário ou pelo dispositivo *IoT* e o escopo enviado através do *token* de acesso são verificados.

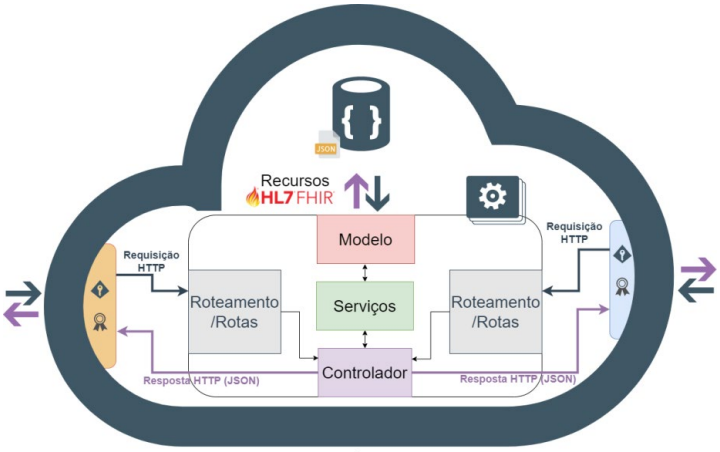

Figura 3 – Arquitetura de software de *H₂Cloud*

O projeto do software *H₂Cloud* está estruturado em blocos de roteamento-controlador-serviços-modelo (Figura 4), conforme as melhores práticas de desenvolvimento de aplicações REST. O bloco de *Roteamento* é responsável por disponibilizar os *endpoints* para as aplicações externas realizar requisições HTTP no servidor *H₂Cloud* e por processar e transmitir os dados das requisições para o bloco *Controlador*. O papel do bloco *Controlador* é selecionar os serviços solicitados pela requisição, verificar o *token* de autenticação e sua validade, retornar um código de sucesso ou erro e enviar um JSON (*JavaScript Object Notation*) com o resultado da requisição. O bloco de *Serviços* é onde está a grande parte da lógica computacional dos fluxos das Figuras 2 e 3, além do controle de erros. O bloco *Modelo* é responsável por modelar e validar que apenas Recursos no padrão FHIR operam leitura e escrita no banco de dados. O banco de dados de *H₂Cloud* é o não-relacional, pois é baseado em coleções de documentos descritas em JSON, onde cada coleção corresponde a um tipo de Recurso FHIR. Quando uma requisição de leitura ou escrita no banco de dados é realizada, basta *H₂Cloud* consultar o tipo do recurso, único atributo obrigatório de qualquer Recurso FHIR, para escolher a coleção que deve ser acessada no banco de dados.

O código-fonte de *H₂Cloud* é desenvolvido como um software de código aberto e disponível para download gratuito num repositório com controle de versão[2]. O servidor está hospedado na nuvem da AWS e os scripts que realizam a implantação de *H₂Cloud* na nuvem também podem ser encontrados no mesmo repositório. A documentação completa dos *endpoints e* as tecnologias empregados no desenvolvimento também estão no repositório.

Resultados e Discussão

Dois experimentos são realizados para comprovar que *H₂Cloud* é capaz de prover interoperabilidade para diferentes tipos de aplicações de saúde que trocam dados por meio de Recursos FHIR e utilizam autenticação *SMART on FHIR*.

Autenticação de Uma Aplicação SMART on FHIR desenvolvida por Terceiros

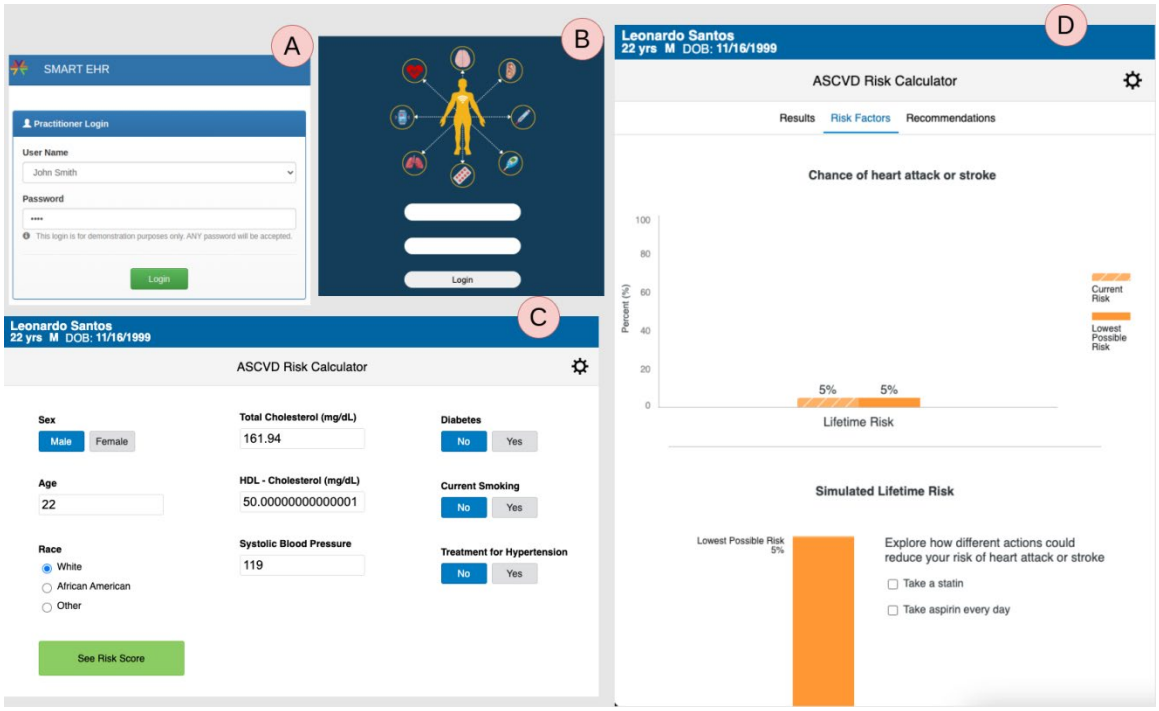

Figura 4 – Capturas de tela da aplicação *ASCVD Risk Calculator* utilizando *H₂Cloud* como servidor ao invés do servidor original. A tela de *login* precisou ser substituída.

[2] https://github.com/if4health/h2cloud

A demonstração que *H₂Cloud* garante interoperabilidade e segurança para aplicações com interface de usuário *SMART on FHIR* consiste na refatoração de uma aplicação desenvolvida por terceiros utilizando *H₂Cloud* como servidor. A aplicação *ASCVD Risk Calculator* foi escolhida porque tem código aberto e gratuito, foi desenvolvida por uma empresa relevante (Cerner, empresa pertencente à Oracle) e não é uma aplicação de teste. *ASCVD Risk Calculator* tem como objetivo calcular o risco de doenças cardíacas de um paciente baseado em alguns fatores como colesterol, pressão sanguínea e fatores sociais. As configurações da aplicação *ASCVD Risk Calculator* são alteradas para realizar todas as requisições em *H₂Cloud*.

Para testar o *ASCVD Risk Calculator.* é preciso clonar o repositório oficial da Cerner[3] e instalar a aplicação e suas dependências na máquina de desenvolvimento local. Em seguida, é necessário substituir a tela de login original (Figura 4-A) da aplicação pela tela de login que direciona a autenticação em *H₂Cloud* (Figura 4-B) e configurar as variáveis de ambiente de *ASCVD Risk Calculator* para utilizar *H₂Cloud* como servidor. Finalmente, algumas adequações também são necessárias para realizar o experimento, pois *ASCVD Risk Calculator* utiliza o FHIR na versão DSTU 2 enquanto *H₂Cloud* suporta a versão R4 do FHIR, a mais recente.

Ao acessar a aplicação, que executa como uma página web, passando como parâmetro o servidor *H₂Cloud*, os processos do SMART ocorrem (fluxo da Figura 2-a), e a tela de login é apresentada (Figura 4-B). Ao entrar com login e senha, as autorizações SMART ocorrerem e a tela inicial da aplicação aparece (Figura 4-C). As informações do formulário (Figura 4-C) são enviadas como Recursos *Patient* e *Observation* do FHIR e são processadas e armazenadas por *H₂Cloud*. Finalmente, a aplicação executa os algoritmos para calcular o risco de ataque cardíaco e apresenta para o usuário (Figura 6-D).

A Figura 5 mostra os Recursos FHIR *Observation* e *Patient* extraídos em formato JSON do banco de dados de *H₂Cloud* após a execução da aplicação *ASCVD Risk Calculator* conforme as capturas de tela da Figura 4. A extração dos Recursos FHIR em formato JSON com o registro da execução da aplicação *ASCVD Risk Calculator*

[3] https://github.com/cerner/ascvd-risk-calculator

comprova o caráter interoperável de *H₂Cloud*, pois não seria possível extraí-los mantendo o servidor original.

Figura 5 –Recursos FHIR Observation e *Patient* em JSON armazenados em *H₂Cloud* e gerados pela aplicação *ASCVD Risk Calculator* na captura de tela da Figura 4-C.

Autenticação com dispositivo IoT

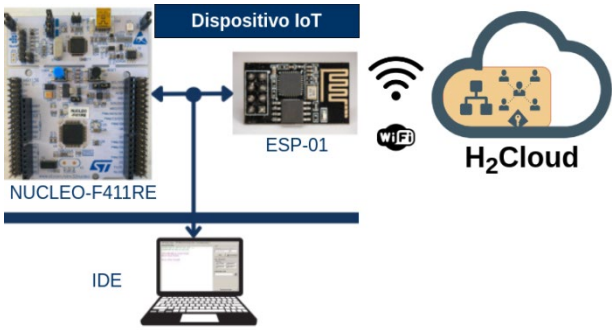

Figura 6 – Diagrama de blocos do *setup* experimental

O objetivo deste experimento é testar e validar a autenticação de uma Aplicação *IoT* no servidor *H₂Cloud*. Compõem o *setup* do experimento (Figura 6) a plataforma de desenvolvimento NUCLEO-F411RE[16] e um módulo ESP-01 para comunicação com internet wi-fi. A comunicação entre dispositivo *IoT* e *H₂Cloud* é monitorada no computador com um ambiente integrado de desenvolvimento (*IDE – integrated development*

environment) para permitir a validação do experimento. Esse é um kit típico para o desenvolvimento de protótipos de aplicações *IoT*.

O primeiro passo do experimento é feito pelo administrador do *H2Cloud* ao cadastrar o dispositivo *IoT* no banco de dados antes do primeiro acesso. Nesse processo são geradas as credenciais do dispositivo ("`client_id`" e "`code`" na Figura 3). Essas informações devem ser inseridas no *firmware* do dispositivo *IoT* e não podem ser expostas para evitar que dispositivos não autorizados acessem o servidor. Após esta configuração permanente de credenciais, basta programar no *firmware do* dispositivo *IoT* o fluxo de autenticação baseado em *SMART on FHIR* (Figura 3).

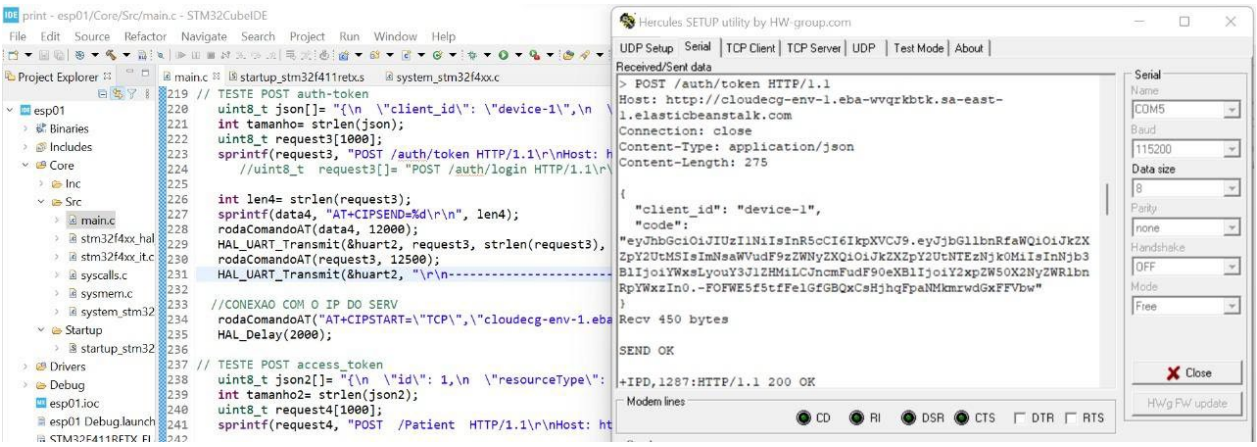

Figura 7 – IDE de programação do firmware do dispositivo *IoT* (esquerda) e janela de monitoramento da comunicação entre dispositivo *IoT* e *H2Cloud* (direita).

A Figura 7 apresenta a captura de tela da IDE (esquerda) com um trecho de código em linguagem C do *firmware* implementado para o experimento. Além disso, é mostrado também o programa Terminal da IDE (direita), onde é possível monitorar a comunicação entre o Dispositivo *IoT* e o servidor *H2Cloud*. Ainda na janela do Terminal (Figura 7), é possível ver uma requisição POST do dispositivo *IoT* ao *H2Cloud* com as credenciais cadastradas no *firmware* do dispositivo *IoT*. O `client_id` cadastrado foi `device-1` e o `code` é um código criptografado. A resposta do *H2Cloud* foi o código 200 (OK), confirmando que os dados de acesso estão de acordo com os previamente cadastrados no banco de dados. Caso o código fosse invalido por qualquer motivo, *H2Cloud* retornaria o código 401 (*unauthorized*) para indicar erro de autenticação.

Conclusão

Este trabalho apresentou *H₂Cloud*, um servidor em nuvem compatível com o padrão FHIR para aplicações de saúde heterogêneas. Para garantir a segurança dos dados em *H₂Cloud*, todas as aplicações devem ser autenticadas utilizando *SMART on FHIR*, fluxo de autenticação recomendado pela organização HL7. *H₂Cloud* foi desenvolvido utilizando as mais modernas práticas de desenvolvimento web. No primeiro experimento, uma aplicação de terceiro – *ASCVD Risk Calculator* – foi modificada para utilizar *H₂Cloud* ao invés do servidor original. No segundo experimento, um protótipo de dispositivo *IoT* sem interface de usuário também consegue trocar recursos FHIR com *H₂Cloud*. Logo, fica comprovado o caráter interoperável e seguro do servidor proposto *H₂Cloud* para um ecossistema heterogêneo de aplicações de saúde. Todos os códigos-fonte desenvolvidos neste trabalho estão disponíveis em repositórios públicos e gratuitos. A iniciativa desenvolvida neste trabalho está alinhada com o plano de ação da ESDB e respeita aspectos éticos e legais. A sexta e a sétima prioridades do plano de ação da ESDB definem: ambiente de interconectividade e ecossistema de inovação, visando a promoção da interoperabilidade com sistemas externos como grande objetivo.

Referências

1. Benson T, Grieve G. Principles of health interoperability: SNOMED CT, HL7 and FHIR. Springer; 2016 Jun 22.
2. FHIR Release 4 [Internet]. HL7.org. 2022. [Acessado em 15/08/2022]. Disponível em: https://www.hl7.org/fhir/
3. Fantonelli M, Celuppi IC, de Oliveira FM, Burigo F, Dalmarco EM, Wazlawick RS. Lei geral de proteção de dados e a interoperabilidade na saúde pública. Journal of Health Informatics. 2021 Mar 15;12.
4. Mandel JC, Kreda DA, Mandl KD, Kohane IS, Ramoni RB. SMART on FHIR: a standards-based, interoperable apps platform for electronic health records. Journal of the American Medical Informatics Association. 2016 Sep 1;23(5):899-908.
5. Rosa M, Faria C, Barbosa AM, Caravau H, Rosa AF, Rocha NP. A fast healthcare interoperability resources (FHIR) implementation integrating complex security mechanisms. Procedia Computer Science. 2019 Jan 1;164:524-31.

6. Barbosa P, Santos D, Lucena C, Bastida L, Ferreira JF, Cea G. The OCARIoT Data Acquisition App. In2020 IEEE 33rd International Symposium on Computer-Based Medical Systems (CBMS) 2020 Jul 28 (pp. 409-414). IEEE.

7. Bloomfield Jr RA, Polo-Wood F, Mandel JC, Mandl KD. Opening the Duke electronic health record to apps: implementing SMART on FHIR. InternationalJournalof Medical Informatics. 2017 Mar 1;99:1-0.

8. Szabó Z, Bilicki V. Access Control of EHR Records in a Heterogeneous Cloud Infrastructure. Acta Cybernetica. 2021 Dec 7;25(2):485-516.

9. Setyawan R, Hidayanto AN, Sensuse DI, Suryono RR, Abilowo K. Data Integration and Interoperability Problems of HL7 FHIR Implementation and Potential Solutions: A Systematic Literature Review. In2021 5th International Conference on Informatics and Computational Sciences (ICICoS) 2021 Nov 24 (pp. 293-298). IEEE.

10. Ayaz M, Pasha MF, Alzahrani MY, Budiarto R, Stiawan D. The Fast Health Interoperability Resources (FHIR) standard: systematic literature review of implementations, applications, challenges and opportunities. JMIR medical informatics. 2021 Jul 30;9(7):e21929.

11. Wagholikar KB, Mandel JC, Klann JG, Wattanasin N, Mendis M, Chute CG, Mandl KD, Murphy SN. SMART-on-FHIR implemented over i2b2. Journal of the American Medical Informatics Association. 2017 Mar 1;24(2):398-402.

12. Kondylakis H, Petrakis Y, Leivadaros S, Iatraki G, Katehakis D. Using XDS and FHIR to support mobile access to EHR information through personal health apps. In2019 IEEE 32nd International Symposium on Computer-Based Medical Systems (CBMS) 2019 Jun 5 (pp. 241-244). IEEE.

13. Ahmad A, Azam F, Anwar MW. Implementation of SMART on FHIR in developing countries through SFPBRF. InProceedings of the 2018 5th International Conference on Biomedical and Bioinformatics Engineering 2018 Nov 12 (pp. 137-144).

14. Mandl KD, Gottlieb D, Ellis A. Beyond one-off integrations: a commercial, substitutable, reusable, standards-based, electronic health record–connected app. Journal of Medical Internet Research. 2019 Feb 1;21(2):e12902.

15. Estratégia de Saúde Digital para o Brasil 2020-2028 [recurso eletrônico] Brasília : Ministério da Saúde, 2020. [Acessado em 09/09/2022]. Disponível em: https://bvsms.saude.gov.br/bvs/publicacoes/estrategia_saude_digital_Brasil.pdf

16. STM32 Nucleo-64 development board with STM32F411RE MCU, supports Arduino and ST morpho connectivity. [Internet]. STMicroelectronics. 2022. [Acessado em 15/08/22]. Disponível em: https://www.st.com/en/evaluation-tools/nucleo-f411re.html