Autenticação Federada para IoT Aplicada a um Sistema Automático de Estacionamento

Maria L. B. A. Santos[1], Jéssica C. Carneiro[1], Antônio M. R. Franco[1], Fernando A. Teixeira[2], Marco A. A. Henriques[3], Leonardo B. Oliveira[1]

[1]UFMG, Belo Horizonte. [2]UFSJ, Ouro Branco. [3]Unicamp, Campinas.

{mburga,jessicarneiro,franco}@dcc.ufmg.br

teixeira@ufsj.edu.br,marco@dca.fee.unicamp.br,leob@dcc.ufmg.br

Abstract. *One of the biggest challenges in the Internet of Things is to provide authentication to devices, especially considering the resource restriction and the potential mobility between different domains of these devices. We present a demonstration of FLAT, our federated authentication solution specially modeled to the IoT environment. In our demo, we apply FLAT to a real scenario: an automatic parking system. The use of only symmetric cryptographic primitives in the IoT Client and implicit certificates make FLAT a lightweight tool to the IoT environment, able to provide security and privacy in device authentication. This demonstration accompanies the paper "FLAT: A Federated Authentication Protocol for the Internet of Things", published in SBRC 2018.*

Resumo. *Um dos grandes desafios em IoT é fornecer autenticação para os dispositivos, especialmente considerando suas restrições de recursos e sua potencial mobilidade entre diferentes domínios. Apresentamos uma demonstração do FLAT, nossa solução de autenticação federada especialmente modelada para IoT. Em nossa demonstração, aplicamos o FLAT a um cenário real de aplicação: um sistema automático de estacionamento. O uso de apenas criptografia simétrica no Cliente IoT e de certificados implícitos tornam o FLAT uma ferramenta leve, fornecendo segurança e privacidade na autenticação de dispositivos IoT. Esta demonstração acompanha o artigo "FLAT: Um Protocolo de Autenticação Federada para a Internet das Coisas", publicado no SBRC 2018.*

1. Introdução

O crescimento da Internet das Coisas (*Internet of Things – IoT*) [Atzori et al. 2010] tem gerado um impacto significativo na sociedade, com inúmeras possibilidades de aplicação nas mais diversas áreas. Este crescimento vem acompanhado de inúmeros desafios, em especial na área de Gestão de Identidade (*Identity Management – IdM*).

IdM refere-se à criação, gerenciamento, uso e exclusão de registros (que identificam pessoas ou coisas) de forma a permitir/autorizar determinadas ações [Windley 2005]. De forma geral, IdM refere-se à identificação de usuários e seu respectivo controle de acesso aos recursos disponíveis em um sistema.

A Gestão de Identidade Federada (*Federated Identity Management – FIdM*) permite que usuários externos se autentiquem em um servidor local sem a necessidade de

criação de novas identidades naquele domínio. No FIdM, o Provedor de Serviços (*Service Provider – SP*) delega a tarefa de autenticação ao Provedor de Identidades (*Identity Provider – IdP*) do domínio de origem do usuário. FIdM permite o controle de acesso de usuários externos a um domínio [Shim et al. 2005]; torna possível o uso de *Single-Sign-On (SSO)*; protege a privacidade do usuário, limitando o armazenamento de informações a apenas uma entidade (seu IdP de origem); e proporciona mais usabilidade ao sistema evitando a necessidade de novos registros/identidades em diferentes SPs.

Abordagens amplamente utilizadas em (F)IdM são inadequadas para a IoT. Primeiramente, nos sistemas existentes os dispositivos utilizam as credenciais de seus respectivos usuários para acessar serviços. Esta prática é insegura, uma vez que os dispositivos normalmente não deveriam ter o mesmo nível de acesso que seus usuários, e muitas vezes não há como atribuir ao dispositivo a identidade de um único usuário (e.x., dispositivos que são utilizados por diversos usuários diferentes). Além disso, ambientes IoT geralmente envolvem um nível maior de mobilidade e interoperabilidade quando comparado a elementos de rede tradicionais [Fremantle et al. 2014]. Por fim, as abordagens existentes fazem uso de criptossistemas caros como o RSA ou DSA, e portanto, requerem um maior uso de recursos computacionais.

De forma a solucionar este problema, propomos o FLAT (*Federated Lightweight Authentication of Things*) [Santos et al. 2018], um modelo federado de autenticação para dispositivos IoT com capacidade de processamento e armazenamento reduzidos. Para produzir uma solução leve, substituímos protocolos e primitivas criptográficas assimétricas por estratégias mais adequadas à IoT. Especificamente, utilizamos uma combinação de certificados implícitos [Brown et al. 2001] e criptografia simétrica para obter uma solução adaptada para dispositivos restritos.

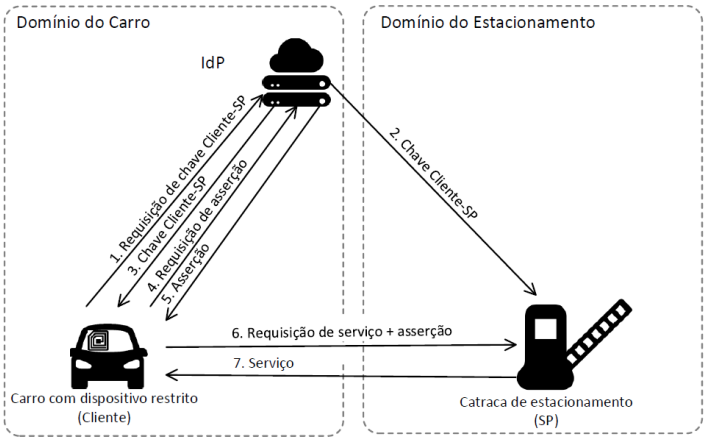


Figura 1: **Exemplo de uso do FLAT em estacionamentos.**

Apresentamos uma demonstração de um cenário de aplicação do protocolo FLAT em que um veículo equipado com um dispositivo restrito (Cliente IoT) de uma universidade precisa se autenticar em uma catraca de entrada do estacionamento (SP) de outra universidade, de forma a poder acessar as vagas disponíveis de um prédio (Fig. 1). O processo de autenticação entre eles é mediado pelo IdP da universidade de origem do veículo, através do protocolo de autenticação FLAT. O Cliente IoT solicita ao IdP uma chave simétrica para comunicação com o SP (passo 1, Fig. 1). O IdP então, envia ao SP e

ao Cliente IoT a chave para que a comunicação entre eles ocorra (passos 2 e 3, Fig. 1). Em seguida, o Cliente IoT solicita uma asserção ao IdP, de forma que seja possível utilizar os serviços oferecidos pelo SP (passo 4, Fig. 1). No passo 5 da Fig. 1, o Cliente IoT recebe a asserção solicitada. De posse da asserção e da chave de comunicação, o Cliente IoT solicita o serviço ao SP (passo 6, Fig. 1), recebendo-o em seguida no passo 7 da Fig. 1.

O restante deste artigo está organizado da seguinte forma: na Seção 2 é apresentado o protocolo de autenticação FLAT e suas premissas. A Seção 3 descreve a demonstração proposta. Na Seção 4 são apresentados os resultados da avaliação de uso de recursos do Cliente IoT pelo FLAT. A Seção 5 discute trabalhos relacionados. Por fim, apresentamos uma conclusão na Seção 6.

2. Autenticação Federada para IoT

Nesta seção apresentamos o FLAT, nosso protocolo de autenticação federada para IoT, além das premissas necessárias ao seu funcionamento.

Pré-implantação de chaves – Assumimos que antes da operação do protocolo de autenticação, há a distribuição prévia de uma chave criptográfica entre o IdP e o Cliente IoT, de forma que eles possam se comunicar utilizando criptografia simétrica posteriormente. A geração desta chave ocorre no Cliente IoT por meio de PUFs (*Physical Unclonable Functions*), conforme descrito em [Suh and Devadas 2007], em que o *hash* da saída do PUF corrigida é utilizada como chave do dispositivo.

Estabelecimento de Confiança – Para que federações distintas possam se comunicar, é necessário o estabelecimento de confiança entre elas. Assumimos que previamente à operação do FLAT há um estabelecimento de confiança entre as federações, baseado no modelo utilizado no eduGAIN [Lopez et al. 2006], em que através de uma Autoridade Certificadora (AC) raiz comum a todas as federações, é estabelecida uma relação de confiança entre cada uma das ACs das federações e a AC raiz, sem a necessidade de uma relação de confiança estabelecida par-a-par entre as ACs de cada federação.

Descoberta de Serviço – Antes da operação efetiva do FLAT, para que o Cliente IoT tenha conhecimento do SP e dos serviços oferecidos por ele, é necessária a descoberta de serviço. Para isso, consideramos que anteriormente à operação do FLAT, o Cliente IoT obtém estas informações através de *beacons*, mensagens de *broadcast* transmitidas periodicamente em um determinado raio de distância, enviados pelo SP.

Figura 2: **Formato de mensagem utilizado no FLAT.**

Formato de Mensagem – O formato das mensagens no FLAT é mostrado na Fig. 2. FLAT utiliza 10 tipos diferentes de mensagens, identificadas no primeiro *byte*. O segundo *byte* identifica o número de sequência. Os próximos 2 campos de 2 *bytes* são utilizados para identificar a origem e o destino da mensagem, respectivamente. Em seguida, temos um campo de 2 *bytes* para o tamanho do *payload* da mensagem. Por fim, definimos um *payload* de até 280 *bytes*, suficiente para armazenar as mensagens mais longas do FLAT (envio de certificados).

Operação do Protocolo – A Fig. 3 mostra a operação do protocolo de autenticação FLAT, utilizando a nomenclatura definida na Tabela 1.

Tabela 1: **Nomenclatura utilizada na descrição da operação do FLAT.**

Client : Cliente IoT	| : operação de concatenação
IdP: IdP do domínio de origem do Cliente	= : operação de atribuição
SP : SP de domínio estrangeiro	n_x: *nonce* gerado por x
P_x: chave pública de x	S_x: chave privada de x
k_x/y: chave secreta k compartilhada entre x e y	cert_x: certificado de x
assert_req: requisição de *assert*	key_req: requisição de chave
serv: serviço propriamente dito	'serv ': descrição do serviço
function(k,s): função sobre a *string* s usando a chave k	

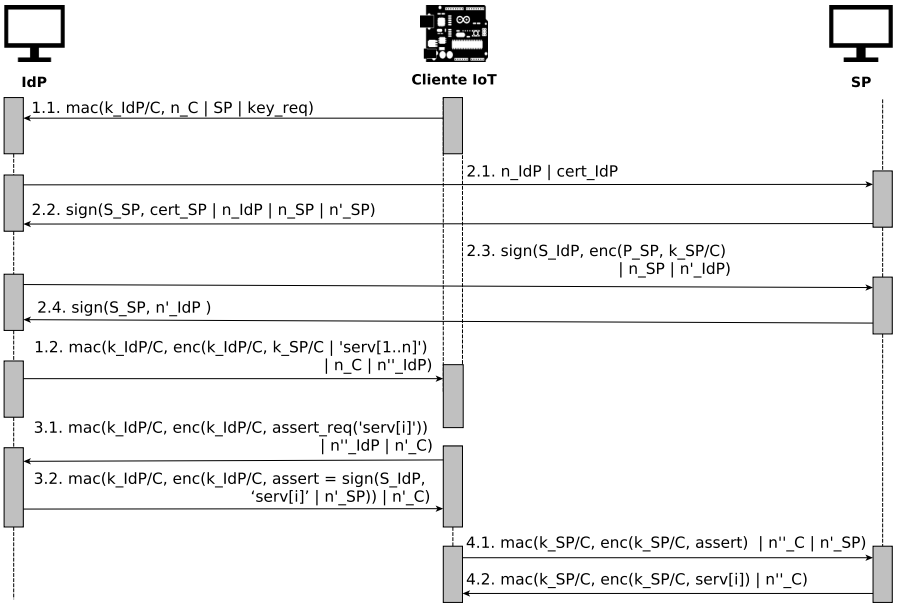


Figura 3: **Protocolo de autenticação FLAT.**

Para que o Cliente IoT utilize os serviços do SP, é necessária a obtenção de uma chave simétrica para proteger a comunicação entre SP e Cliente IoT, fornecida pelo IdP. Dessa forma, no passo 1.1 da Fig. 3, o Cliente IoT solicita esta chave ao IdP.

Em seguida, o IdP inicia a comunicação com o SP, estabelecendo um canal seguro para envio da chave (passos 2.1 e 2.2, Fig. 3), que é enviada no passo 2.3 da Fig. 3. O SP então confirma o recebimento da chave no passo 2.4 da Fig. 3.

No passo 1.2 da Fig. 3 a requisição de chave do Cliente IoT ao IdP é concluída, com o envio da chave simétrica de comunicação também ao Cliente IoT.

O Cliente IoT então solicita e recebe do IdP uma asserção (*assert*), de forma a obter acesso ao serviço do SP (passos 3.1 e 3.2, Fig. 3).

Por fim, com a chave e a asserção, o Cliente IoT solicita o serviço (passo 4.1, Fig. 3), recebendo em seguida a resposta do SP com o serviço (passo 4.2, Fig. 3).

3. Demonstração

Dentre os diversos cenários que requerem autenticação de dispositivos de diferentes domínios, escolhemos para a demonstração do FLAT um sistema automático de estacionamento em universidades.

Considere que um veículo pertencente à Universidade Federal de São João Del-Rei (UFSJ) esteja em uma visita técnica à Universidade Federal de Minas Gerais (UFMG), e deseje utilizar o estacionamento de um dos prédios da UFMG, como o do Instituto de Ciências Exatas (ICEx). Admitindo que exista um estabelecimento de confiança prévio entre as universidades, este veículo pode, utilizando o protocolo de autenticação FLAT, se autenticar e utilizar o estacionamento do ICEx. Em nossa demonstração (Fig. 4), um

Figura 4: **Demonstração do FLAT: sistema automático de estacionamento.**

veículo (Cliente IoT) se aproxima de uma catraca de estacionamento (SP). Através de seu IdP de origem, o veículo irá se autenticar para utilizar o serviço. Caso a autenticação seja bem sucedida, o SP irá conceder o serviço ao Cliente IoT, acendendo uma luz de LED verde e abrindo a catraca. Caso contrário, uma luz de LED vermelha é acesa e a catraca permanece fechada, impedindo o acesso do veículo ao estacionamento.

Utilizamos em nosso modelo uma maquete contemplando um prédio com estacionamento, uma catraca e carros em miniatura. A catraca representando o SP está conectada a um servo motor Tower Pro 9G SG90 controlado por um Raspberry Pi Zero W (Broadcom BCM2385 ARM1176JZF-S 1GHz, 512 MB RAM, 8GB microSD), que também é responsável pelas luzes de LED vermelha e verde. A energia é fornecida ao Raspberry por meio de um adaptador conectado a um ponto de energia elétrica. Os carros em miniatura possuem um dispositivo Arduino Due (Atmel 84 MHz, 96 kB SRAM, 512 kB flash) com *shields* Arduino Wi-Fi (802.11 b/g) acoplados a eles, e recebem energia de uma bateria conectada ao Arduino. O IdP é representado por um laptop (Intel Core i7 2,7 GHz, 8GB RAM, 1TB HD). A comunicação entre os dispositivos ocorre por meio de rede sem fio local (802.11 b/g), fornecida por um ponto de acesso (MikroTik hAP ac lite).

3.1. Implementação e Arquitetura

A implementação do FLAT foi desenvolvida em C/C++ utilizando as bibliotecas Relic [Aranha and Gouvêa] (criptografia) e Arduino Wi-Fi[1] (comunicação). Também foi utilizado um *Makefile*[2] adaptado para auxiliar na depuração e execução do código. Manuais de utilização, código, vídeos explicativos, licenças de utilização e demais informações

[1] `https://www.arduino.cc/en/Reference/WiFi`
[2] `https://github.com/pauldreik/arduino-due-makefile`

sobre o FLAT podem ser encontradas na página: `https://sites.google.com/view/flat`.

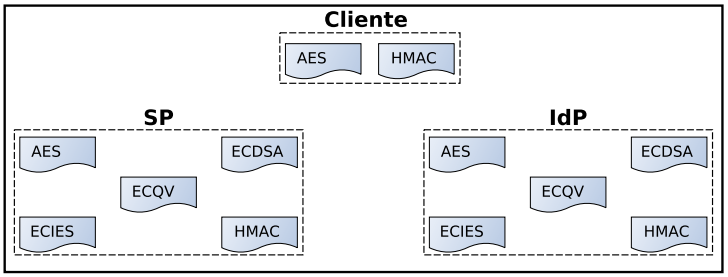


Figura 5: **Módulos criptográficos do FLAT.**

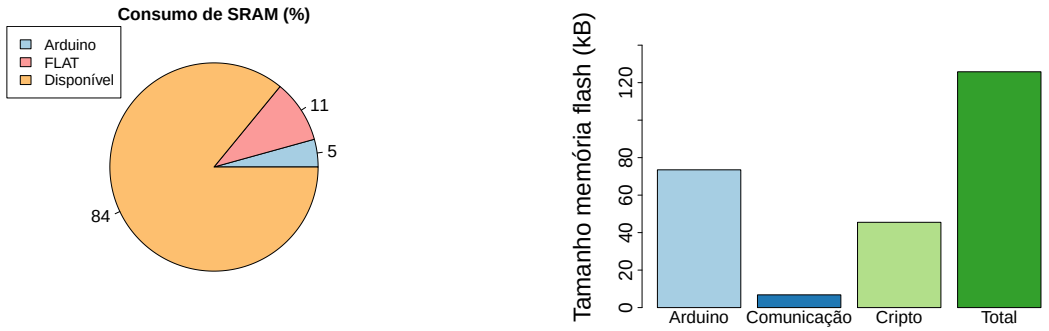
A Fig. 5 mostra os módulos criptográficos utilizados no FLAT. No Cliente IoT foram utilizados apenas criptossistemas simétricos (AES para cifração simétrica e HMAC para MACs). No SP e IdP foram utilizados tanto módulos simétricos quanto assimétricos: AES para cifração simétrica, HMAC para MACs, ECIES para cifração assimétrica, ECDSA para assinatura digital e ECQV para certificados implícitos.

A substituição de certificados tradicionais por certificados implícitos (ECQV), criptografia assimétrica tradicional (RSA/DSA) por criptografia assimétrica baseada em curvas elípticas (ECIES/ECDSA) e o uso de apenas primitivas criptográficas simétricas no Cliente IoT, tornam o FLAT um protocolo mais leve para o ambiente IoT.

4. Resultados

Avaliamos o FLAT em termos de consumo de SRAM, armazenamento e comunicação, considerando o dispositivo mais restrito de nossa solução, o Cliente IoT (Arduino Due).

SRAM – Calculamos o consumo de SRAM embarcando no Cliente IoT inicialmente uma aplicação vazia, que apenas calculava o consumo de SRAM, encontrando o valor de 4,8 kB (Arduino, Fig. 6a). Em seguida, o consumo de memória foi calculado através da execução do FLAT no Cliente IoT, obtendo o valor de 10,9 kB. Quando comparado com a SRAM disponível no Arduino Due (96 kB), o uso de SRAM na execução do FLAT juntamente com as bibliotecas nativas do Arduino é de aproximadamente 16%.



(a) Custo de SRAM. **(b) Tamanho do FLAT.**

Figura 6: **Custos do Cliente IoT.**

Armazenamento – Para calcular o uso de memória *flash* do FLAT, inicialmente embarcamos no Cliente IoT uma aplicação vazia, verificando que esta ocupava 73,5 kB (Arduino,

Fig. 6b). Calculamos também o custo de armazenamento das funções de comunicação (Comunicação, Fig. 6b) e de criptografia (Cripto, Fig. 6b) do FLAT, obtendo os valores de 6,8 kB e 45,5 kB, respectivamente. Por fim, calculamos o tamanho total ocupado pelo FLAT juntamente com as bibliotecas nativas do Arduino, obtendo o valor de 125,8 kB (Total, Fig. 6b). Comparando com a memória *flash* disponível no Arduino Due, o custo total de armazenamento do FLAT é de cerca de 24% da memória *flash* disponível.

Comunicação – Comparamos os custos de comunicação do Cliente IoT com um protocolo Baseline genérico baseado no OAuth[3]. O volume de dados trocados com o Cliente IoT no Baseline é cerca de 72% maior que no FLAT (Fig. 7). FLAT elimina a necessidade de uso de criptografia assimétrica e troca de certificados no Cliente IoT, que de forma geral, geram uma grande sobrecarga de comunicação.

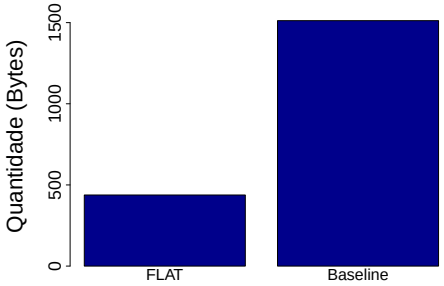


Figura 7: **Volume de comunicação do Cliente IoT.**

5. Trabalhos Relacionados

Existem várias propostas de FIdM para IoT (e.x. [Fremantle and Aziz 2016, Cirani et al. 2015, Domenech et al. 2016]) que se baseiam em soluções de FIdM para a Internet tradicional. Domenech et al. propõem uma solução de FIdM para IoT com suporte ao SAML[4], disponibilizada no GIdLab [Wangham et al. 2013]. Apesar de garantirem a integração com sistemas existentes, por serem soluções tradicionais adaptadas para IoT, estas abordagens exigem dispositivos mais robustos, como o BeagleBone por exemplo. Silva e Silva propõem uma solução de autorização federada para IoT utilizando SAML, RBAC e um controlador lógico com maior poder computacional [Silva and Silva 2017]. FLAT, por sua vez, é uma solução de autenticação que pode ser empregada mesmo em dispositivos IoT com maiores restrições de recursos, sem o uso de *gateways* externos.

Também há iniciativas que abordam soluções para ambientes IoT específicos, como casas inteligentes [Hong et al. 2016] ou aplicações para assistência técnica de equipamentos [Witkovski et al. 2015]. FLAT, no entanto, tem como objetivo ser mais abrangente, podendo ser utilizado nos mais diversos ambientes IoT.

6. Conclusão

Apresentamos uma demonstração do FLAT, um protocolo de autenticação federada especialmente modelado para IoT. Em nossa demonstração, aplicamos o FLAT a um cenário de controle automático de estacionamento em universidades, apresentando detalhes de implementação e arquitetura. Ainda, realizamos uma análise dos custos de comunicação,

[3] `https://oauth.net/`

[4] `https://tools.ietf.org/html/rfc7522`

consumo de SRAM e memória *flash* no Cliente IoT, demonstrando que FLAT pode ser utilizado na autenticação para IoT em ambientes com restrições de recursos.

Agradecimentos

Agradecemos à RNP pelo apoio e financiamento deste projeto. Também agradecemos o apoio de CAPES, CNPq e FAPEMIG.

Referências

Aranha, D. F. and Gouvêa, C. P. L. RELIC is an Efficient LIbrary for Cryptography. `https://github.com/relic-toolkit/relic`.

Atzori, L., Iera, A., and Morabito, G. (2010). The Internet of Things: A survey. *Computer Networks*, 54(15):2787 – 2805.

Brown, D. R., Gallant, R., and Vanstone, S. A. (2001). Provably Secure Implicit Certificate Schemes. In *FC'01*. Springer.

Cirani, S., Picone, M., Gonizzi, P., Veltri, L., and Ferrari, G. (2015). IoT-OAS: An OAuth-based Authorization Service Architecture for Secure Services in IoT Scenarios. *IEEE Sensors Journal*, 15(2):1224–1234.

Domenech, M. C., Boukerche, A., and Wangham, M. S. (2016). An Authentication and Authorization Infrastructure for the Web of Things. In *Q2SWinet*. ACM.

Fremantle, P. and Aziz, B. (2016). OAuthing: Privacy-enhancing Federation for the Internet of Things. In *CIoT'16*. IEEE.

Fremantle, P., Aziz, B., Kopecký, J., and Scott, P. (2014). Federated Identity and Access Management for the Internet of Things. In *SIoT'14*. IEEE.

Hong, J., Levy, A., and Levis, P. (2016). Demo: Building Comprehensible Access Control for the Internet of Things Using Beetle. In *MobiSys'16*. ACM.

Lopez, D. R., Macias, J., Molina, M., Rauschenbach, J., Solberg, A., and Stanica, M. (2006). *Deliverable DJ5.2.3.1: Best Practice Guide – AAI Cookbook*. Géant 2.

Santos, M. L. B. A., Carneiro, J. C., Franco, A. M. R., Teixeira, F. A., Henriques, M. A. A., and Oliveira, L. B. (2018). FLAT: Um Protocolo de Autenticação Federada para a Internet das Coisas. In *SBRC'18*. SBC.

Shim, S. S., Bhalla, G., and Pendyala, V. (2005). Federated Identity Management. *IEEE Computer*, 38(12):120–122.

Silva, C. E. and Silva, G. C. (2017). Uma Proposta de Arquitetura para Autorização Federada com Internet das Coisas. In *SBSeg'17*. SBC.

Suh, G. E. and Devadas, S. (2007). Physical Unclonable Functions for Device Authentication and Secret Key Generation. In *44th DAC*. ACM.

Wangham, M. S., Mello, E. R. d., Souza, M. C., and Coelho, H. (2013). Gidlab: Laboratório de experimentação em gestão de identidades. In *SBSeg'13*. SBC.

Windley, P. J. (2005). *Digital Identity: Unmasking Identity Management Architecture (IMA)*. O'Reilly Media, Inc.

Witkovski, A., Santin, A., Abreu, V., and Marynowski, J. (2015). An IdM and Key-based Authentication Method for Providing Single Sign-On in IoT. In *GLOBECOM*. IEEE.