Autenticação mútua de nós sensores com nós intermediários para IoT no contexto de Fog Computing

Lukas Derner Grüdtner[1], **Leandro Loffi**[1], **Carlos Becker Westphall**[1], **Carla Merkle Westphall**[1]

[1] Laboratório de Redes e Gerência (LRG) – Departamento de Informática e Estatística
Universidade Federal de Santa Catarina (UFSC) – Florianópolis – SC – Brasil

lukasgrudtner@hotmail.com, leandroloffi3@gmail.com,

carlos.westphall@ufsc.br, carla.merkle.westphall@ufsc.br

Abstract. *Several paradigm evolutions have been proposed in recent years. Fog Computing is an area of Computer Science that is under constant development and evolution, and in conjunction with information security, the paradigm becomes more reliable and secure for edge IoT platforms. Security issues are difficult to reach in environments with limited resources. This work aimed to improve an authentication model, taking into account IoT devices in the context of Fog Computing. Finally, validation through the implementation of a mutual authentication system has produced the result of confidentiality, integrity and authenticity, allied with embedded devices used in the Fog Computing paradigm.*

Resumo. *Várias evoluções de paradigmas foram propostas nos últimos anos. Fog Computing é uma área da Ciência da Computação que está em construção e constante evolução, e em conjunto com a segurança da informação, o paradigma se torna mais confiável e seguro para as plataformas da borda do IoT. Os quesitos de segurança são de difícil alcance em ambientes com recursos limitados. Este trabalho teve como objetivo aprimorar um modelo de autenticação, levando em consideração dispositivos IoT no contexto de Fog Computing. Por fim, a validação por meio da implementação de um sistema de autenticação mútua produziu o resultado de confidencialidade, integridade e autenticidade, aliada com dispositivos embarcados utilizados no paradigma Fog Computing.*

1. Introdução

A *Internet of Things* (*IoT*) é um paradigma recente que visa a integração das "coisas", ou objetos, através da rede. A principal característica da *IoT* é ter alguns itens que caracterizam uma rede de sensores e atuadores [Xia et al. 2012]. Em qualquer ambiente poderá haver sensores, os quais capturam e enviam dados para um servidor central, também conhecido como *Cloud*, e esses dados serão processados e utilizados posteriormente em algum determinado serviço [Atzori et al. 2010].

A utilização de *IoT* no contexto de *Cloud Computing* traz à tona a questão do consumo de largura de banda. A utilização da banda por inúmeros dispositivos de borda para a comunicação com a *Cloud* acaba causando grandes congestionamentos na rede. Assim, o paradigma *Fog Computing* surge para resolver esta dificuldade. Em seus trabalhos, Yi et al. [Yi et al. 2015] e Bonomi et al. [Bonomi et al. 2012] conceituam que, neste paradigma, há intermediadores que se posicionam entre o servidor central e a borda da rede,

realizando o processamento de maneira local e, assim, minimizam o congestionamento e a latência no tráfego para a *Cloud*. Os autores ainda mencionam que o paradigma *Fog Computing* resulta em baixa latência na rede, distribuição geográfica ampla, mobilidade, grande número de nós, além da heterogeneidade de dispositivos.

Conceitualmente, a segurança no contexto da informação se refere à proteção dos dados e a preservação de seus valores, a exemplo disso é utilizado a autenticação [Schneier 1995]. Em *IoT*, a autenticação trata de especificar se um dado sensor conectado à rede é realmente quem ele diz ser, dificultando, assim, a personificação. Parte da dificuldade reside na comunicação entre o dispositivo central e o sensor, dado que métodos de autenticação devem ser seguros, mas sem perder de vista as limitações características destes dispositivos. Atualmente, para a autenticação mútua, utilizam-se métodos de troca de chaves, como *Diffie-Hellman* (*DH*) e *RSA* (*Rivest-Shamir-Adleman*), além de criptografia assimétrica e simétrica, para autenticar a origem e o destinatário.

Liu et al., em [Liu et al. 2012], propuseram um protocolo para o estabelecimento de chaves seguras de forma simples e eficiente baseado em curvas elípticas (*Elliptic Curve - EC*) e, além disso, a utilização de uma política de controle de acesso, o qual adota um método baseado em *RBAC* (*Role-Based Access Control*). Para seu propósito de autenticação, os autores Jam et al., [Jan et al. 2014], utilizaram um modelo de interação cliente-servidor baseado em *CoAP* (*Constrained Application Protocol*), onde cada objeto cliente é restrito a uma única conexão com determinado servidor. Para aumentar a eficiência e a robustez, eles adicionaram características de criptografia para a autenticação no *CoAP*.

Em [Rewagad and Pawar 2013], os autores implementaram um modelo de assinatura digital utilizando *Diffie-Hellman Efêmero* (*DHE*) com os algoritmos *RSA* e *AES* (*Advanced Encryption Standard*) aplicados no contexto de *Cloud Computing*. Já em [Kothmayr et al. 2013], foi apresentado uma implementação completa de um esquema de segurança com autenticação baseada no protocolo *DTLS* (*Datagram Transport Layer Security*) e métodos criptográficos utilizando o algoritmo *RSA*.

A autenticação é um dos pontos principais para o bom funcionamento de uma rede [Schneier 1995], podendo também ser aplicada em uma rede de sensores. Cada dispositivo conectado à rede possui um *ID*, o qual é único e o representa dentro daquela rede. Caso os métodos de autenticação utilizados sejam fracos e/ou apresentem falhas, um invasor poderia se personificar como um sensor válido, obtendo, assim, um *ID* para este dispositivo mal-intencionado e tornando-o autenticado dentro do sistema. Ele poderia, portanto, gerar dados falsos ou até mesmo infectar outras partes do sistema com o intuito de obter dados sigilosos e/ou causar mal-funcionamento no sistema.

Este artigo aplica protocolos existentes de segurança para a autenticação mútua de um dispositivo à rede *IoT* no contexto de *Fog Computing*. Para tanto, utilizamos um modelo de protocolo baseado na troca de chaves de *Diffie-Hellman Efêmero*, onde o nó sensor e o nó intermediário trocam chaves para obter uma chave em comum e, posteriormente, manter a comunicação cifrada com o algoritmo de criptografia *AES* de 256 bits com modo *CBC* (*Cipher Block Chaining*). Assim, obtemos a cifragem dos dados trafegados na comunicação, mantendo-a segura. Esta solução, portanto, contempla os princípios de autenticidade, integridade e confidencialidade.

De modo geral, o presente trabalho abordou as seguintes contribuições: a otimização de um (I) Modelo de autenticação mútua com protocolos existentes, (II) Aplicação do modelo para validação, (III) Implantação em um ambiente real de *Fog Computing*.

O trabalho está subdividido em seis seções: na seção 2 é apresentada uma revisão bibliográfica, na seção 3 são apresentados os conceitos básicos, na seção 4 são abordados aspectos relacionados ao modelo otimizado, nas seções 5 e 6 são expostas a implementação e os resultados obtidos e, por fim, na seção 7, são apresentadas as conclusões e trabalhos futuros.

2. Trabalhos Relacionados

Liu et al. [Liu et al. 2012], em seu trabalho, apresentaram uma arquitetura para a troca segura de chaves utilizando métodos baseados em *ECC* (*Elliptic Curve Cryptography*). Neste sistema, cada nó da rede obtém um registro em um ponto de acesso confiável próximo a ele, e isto é chamado de *Registration Authority* (*RA*), e, a partir de uma curta sequência de passos, a troca de chaves entre as entidades é estabelecida. Apesar de abordarem a segurança de sua arquitetura quanto a ataques - como *eavesdropping*, *man-in-the-middle*, *key control* e *replay* – seu artigo é limitado apenas para a autenticação, e não apresenta aspectos relacionados à confidencialidade em sua arquitetura.

Já Jan et al. [Jan et al. 2014] descreveram um algoritmo de autenticação utilizando a interação de cliente-servidor baseada em *CoAP*, onde, cada cliente, ou objeto, é restrito a uma única conexão com determinado servidor. Essa escolha é justificada, pois *CoAP* foi projetado para dispositivos com restrições energéticas e, além disso, a gravidade de uma intrusão é diretamente proporcional ao número de conexões estabelecidas. Seu modelo para a autenticação foi realizado com a troca de chaves baseada em *RSA*, utilizando métodos criptográficos de curvas elípticas. Semelhante ao presente trabalho, porém, este apresenta um modelo utilizando *Diffie-Hellman Efêmero* e confidencialidade com criptografia simétrica.

No trabalho de Rewagad e Pawar [Rewagad and Pawar 2013] foi aplicado um modelo de autenticação de cliente-servidor com os principais pontos de segurança, sendo a autenticidade, integridade e confidencialidade no contexto de *Cloud Computing*. Em seu modelo, foi utilizada criptografia *RSA* para a assinatura dos parâmetros, além da troca de chaves *Diffie-Hellman* e o algoritmo simétrico *AES*. A limitação do modelo proposto por Rewagad e Pawar é a autenticação de apenas uma via, além da não cifragem dos parâmetros com a chave pública do receptor. Por fim, seu modelo é mais adequado para o ambiente de *Cloud Computing*, enquanto o presente trabalho direciona-se para *Fog Computing*.

Por fim, Kothmayr et al. [Kothmayr et al. 2013] definiram um modelo de autenticação em dois passos para *IoT* baseado principalmente no protocolo *DTLS*. Eles incorporaram em sua arquitetura um modelo de segurança de ponta a ponta, dado que, deste modo, é necessária uma carga menor de processamento criptográfico entre os nós, deixando a responsabilidade apenas para os nós das extremidades. Eles introduziram ainda um servidor de controle de acesso (*Access Control - AC*), o qual é uma entidade confiável e detentora dos direitos de acesso à rede.

Deixa-se claro que este trabalho teve por objetivo aplicar protocolos existentes,

com alguma uma variação, para a autenticação mútua de dispositivos com baixa memória. Portanto, não foi considerado, de forma explícita, que há trabalhos que abordam pontualmente o problema de autenticação mútua ou até melhor que este, mas, de modo geral, assuntos semelhantes e superficiais, para dar embasamento ao presente trabalho.

Na Tabela 1 é apresentada uma comparação do presente trabalho com os trabalhos abordados anteriormente. De forma simples, nota-se que o presente trabalho contemplou todo o conteúdo, além da troca de chaves mútua.

Tabela 1. Comparação entre os trabalhos

Trabalho	Paradigma	Autenticidade	Confidencialidade
[Liu et al. 2012]	*IoT*	*EC_OpenID*	-
[Jan et al. 2014]	*IoT*	*EC_RSA*	*AES128_CBC*
[Rewagad and Pawar 2013]	*Cloud Computing*	*DHE_RSA*	*AES128_CBC*
[Kothmayr et al. 2013]	*IoT*	*DHE_RSA*	Depende Handshake
Este Trabalho	*Fog Computing*	*DHE_RSA*	*AES256_CBC*

3. Conceitos Básicos

A Internet das Coisas, ou *Internet of Things*, é um conceito que nos permite adentrar em diversos domínios onde sequer imaginava-se que a tecnologia contaria com alguma utilidade. Conceitos tais como *smart cities*, *smart environment*, *smart metering*, dentre muitos outros, estão aí para comprovar o quão enraizada está a *IoT* em nosso cotidiano e em nosso mundo.

Os sensores são os responsáveis por captar sinais do ambiente, transformando-os em dados que, por meio da rede, comunicam-se entre si e provêm informações que nos permite reduzir desperdícios, custos e perdas, além de obter otimizações em qualquer área de nosso interesse [Xia et al. 2012]. Na Figura 1, é possível visualizar as camadas entre os nós sensores, nós intermediários, Internet e Computação em Nuvem. No entanto, o presente trabalho foca apenas na autenticação dos nós sensores na camada do nó intermediário.

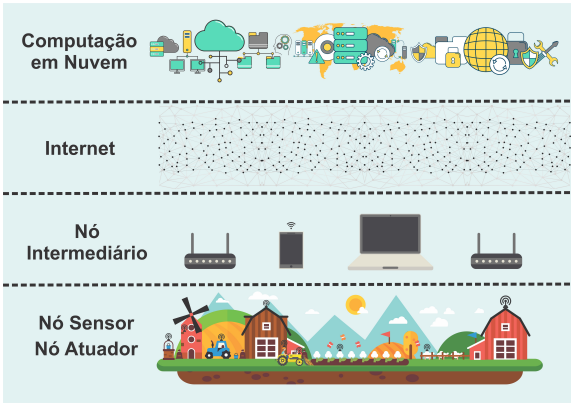

Figura 1. A divisão das camadas no paradigma IoT

Os dados coletados pelos sensores são enviados para um servidor central, também chamado de *Cloud*, depois processados e então as informações tomam um significado que atende nossos interesses. Porém, este processo acarreta em um grande congestionamento

na rede. Foi pensando nisto que o paradigma de Computação em Nevoeiro, ou *Fog Computing*, obteve sua formulação. Neste paradigma, existem dispositivos intermediários entre os sensores e a *Cloud*, os quais são concentradores que realizam um pré-processamento dos dados antes de enviá-los à *Cloud*. Ou seja, *Fog Computing* instituiu uma nova camada na rede com a finalidade de reduzir o tráfego [Bonomi et al. 2012].

4. Autenticação mútua

A autenticação dos nós sensores foi dividida em alguns passos: a descoberta de dispositivos, a troca de chaves, a autenticação e a criptografia dos dados. Considerando apenas duas partes, o CLIENTE A (um nó sensor qualquer) e o SERVIDOR B (um *gateway* ou nó intermediário), a comunicação dá-se em passos, e cada um deles pode ser visto com mais detalhes na Figura 2. Destaca-se que a autenticação dos nós sensores efetua a troca de chaves empregando *Diffie-Hellman Efêmero* para a obtenção de uma chave de sessão ou chave secreta.

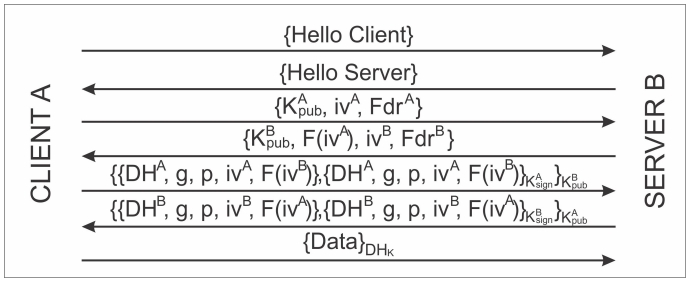

Figura 2. Autenticação dos nós sensores ao Gateway

O primeiro passo inicia-se com o CLIENTE A enviando um pacote para o SERVIDOR B, com o objetivo de criar uma "conexão". Para fins de visualização, este pacote contém a cadeia de caracteres "*HELLO CLIENT*". O segundo passo dá-se com a resposta do SERVIDOR B para o CLIENTE A com a cadeia "*HELLO SERVER*". Com isso, a conexão é estabelecida. Para um melhor desempenho, sugere-se a utilização de cadeias de *bits* para o estabelecimento da conexão.

No terceiro passo, o CLIENTE A gera um par de chaves assimétricas, sendo elas uma pública (K^A_{pub}) e a outra privada (K^A_{priv}). Para a geração, é necessário um *Initialization Vector* (*IV*) com valores aleatórios, o que garante a distinção entre as chaves geradas. Então, um pacote é enviado ao SERVIDOR B contendo: a chave pública do CLIENTE A (K^A_{pub}); um valor como "desafio-resposta", gerado pelo próprio CLIENTE A; e uma cadeia de caracteres *Fdr*, a qual define qual é a função do "desafio resposta".

No quarto passo, o SERVIDOR B gera um par de chaves assimétricas, sendo elas uma pública (K^B_{pub}) e a outra privada (K^B_{priv}). Na sequência, o SERVIDOR B recebe o pacote do CLIENTE A, e responde com outro pacote contendo sua chave pública (K^B_{pub}) e a resposta para o "desafio-resposta", calculado a partir da função *Fdr* – uma função matemática predefinida, podendo ser soma, subtração ou multiplicação – aplicada ao valor de *IV* recebido , o que pode ser visualizado na etapa 3 da Figura 2.

No quinto passo, o CLIENTE A realiza o cálculo dos valores de *Diffie-Hellman*. Um novo pacote constituído pelo valor *DH* obtido (DH^A), os parâmetros g e p utilizados no cálculo, um novo valor de *IV* (iv^A) e o valor de *IV* obtido do SERVIDOR B aplicado

a função *Fdr* ($F(iv^B)$) será enviado para o SERVIDOR B. Além disso, é aplicada uma função de resumo (*Hash*) para todos estes dados, e o seu resultado é cifrado com a chave privada do CLIENT A (K^A_{priv}) e então incluído no pacote. Agora, todo o pacote é cifrado com a chave pública do SERVIDOR B (K^B_{pub}). A cifragem resulta na resguarda da garantia da confidencialidade dos dados.

No sexto e último passo da troca de chaves, o SERVIDOR B realiza o cálculo dos valores de *Diffie-Hellman* a partir das informações provenientes do CLIENTE A. O SERVIDOR B, então, realiza as mesmas ações tal como feito pelo CLIENTE A no passo anterior, e, ao final do processo, envia o pacote resultante para o CLIENTE A. Com isso, ambas as partes possuem uma chave em comum: a chave de sessão (DH_K).

Após a troca de chaves, o cliente e o servidor serão capazes de trocar dados cifrados com uma chave simétrica (DH_K), que pode durar tanto por apenas uma sessão como também ser permanente. Outro autenticador de emissor e receptor é o valor de *IV*, onde toda e qualquer troca de mensagens necessitará ter conhecimento da função definida (*Fdr*) no início da troca de chaves.

5. Metodologia e Desenvolvimento

A implementação do método foi desenvolvida utilizando-se as linguagens C e C++. Para a modelagem e desenvolvimento do CLIENTE A, foi utilizada a *IDE* (*Integrated Development Environment*) do *Arduino.cc*, enquanto que o SERVIDOR A foi desenvolvido utilizando-se um editor de texto e um terminal no sistema operacional *Ubuntu*.

As bibliotecas externas que foram utilizadas como auxílio no desenvolvimento dos códigos-fontes do CLIENTE A são: AES.h, SPI.h, Ethernet.h, EthernetUdp.h, math.h, string.h, printf.h. Já para o SERVIDOR B, foram: stdio.h, stdlib.h, string.h, unistd.h, netdb.h, sys/types.h, sys/socket.h, netinet/in.h, iostream, sstream, cmath, string e AES.h.

O código-fonte do método de autenticação mútua está disponível no *GitHub*, e pode ser acessado através do link:

`https://github.com/lukasgrudtner/iotAuth`

Para a implantação do método, foi utilizada uma placa Arduino do modelo Uno, juntamente com uma *Shield Ethernet* conectada em um *switch* local. Para o primeiro teste, utilizou-se de um servidor, sendo ele um computador conectado a este mesmo *switch*. Em um segundo teste, foi utilizada um outro computador remoto para simular a *Cloud*. Em resumo, a topologia utilizada pode ser visualizada na Figura 3.

6. Resultados

Os resultados apresentados aqui se relacionam com o objetivo do trabalho, que é aplicar a autenticação mútua de nós sensores em ambiente *IoT* com contexto em *Fog Computing*. Em relação a autenticação mútua, o modelo utiliza uma função "desafio-resposta", na qual apenas quem a recebe tem condições para responder ao desafio, e ambas as partes deverão respondê-la a cada interação. Outra forma composta de autenticação utilizada no modelo é através da criptografia assimétrica. Para manter-se leve, o modelo abordou somente a questão de segurança de autenticação mútua e não incluiu a verificação da legitimidade do nó perante a cadeia de certificação. Por este motivo, qualquer cliente é capaz de se

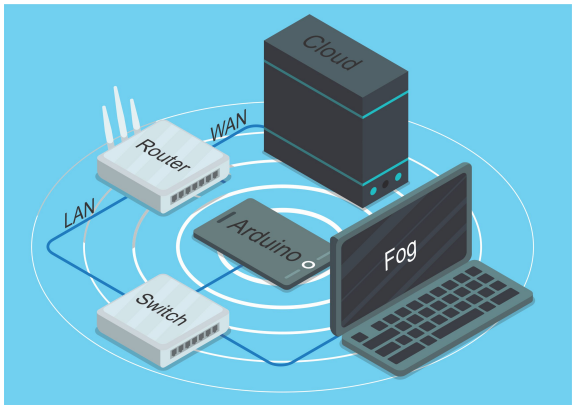

Figura 3. Topologia da Rede

autenticar ao servidor, e vice-versa. Caso alguma das duas partes falhe, a autenticação mútua falha totalmente.

O método de autenticação foi aplicado em dois dispositivos: uma placa Arduino e um computador. No Arduino, foram utilizados 17.470 *bytes* para o armazenamento do código-fonte, o que compreende cerca de 54% do espaço disponível. Já na memória dinâmica, 1.109 *bytes* foram alocados para variáveis globais, deixando 939 *bytes* para uso de variáveis locais. Um ponto que é importante destacar é que pôde-se acompanhar todos os passos que foram citados na Figura 2 através da porta serial do Arduino e na saída do terminal do computador. Ambos os resultados são apresentados na Figura 4.

CLIENT A

```
**********HELLO CLIENT**********
Hello Client: Successful
**********HELLO SERVER**********
Server Client: Successful
**********SEND RSA CLIENT**********
RSA Public Key: 9827
IV: 8
**********RECEIVED RSA SERVER**********
RSA Public Key: 8736
IV: 9
```

```
**********SEND DH CLIENT**********
Diffie-Hellman Key: 13
g: 23
p: 86
IV: 8
*********RECEIVED DH SERVER*********
Diffie-Hellman Key: 41
IV: 9
***SYMMETRICAL SESSION CLIENT-SERVER***
Session Key: 47
```

SERVER B

```
******HELLO CLIENT AND SERVER******
Hello Client and Server Successful!
******CLIENT RSA KEY RECEIVED******
Client RSA Public Key: 9827
IV: 8
FDR: IV (8) + 1
*******SEND SERVER RSA KEY*********
Sent Message: 8736#9
Client and Server RSA KEY Successful!
Server RSA Public Key: 8736
IV Obtained: 9
```

```
*******CLIENT DH KEY RECEIVED******
Diffie-Hellman Key: 13
Base: 23
Modulus: 86
Client IV: 8
Session Key: 47
*********SEND SERVER DH KEY********
Sent Message: 41#9
Client and Server DH KEY Successful!
Diffie-Hellman Key: 47
*SYMMETRICAL SESSION CLIENT-SERVER*
Session Key: 47
```

Figura 4. Tela de saída da troca de chaves no CLIENT A e SERVER B

No primeiro teste, com cliente e servidor situados na mesma rede local, obtivemos uma média de 1000 ms para a autenticação mútua. Em um segundo teste, com cliente e

servidor situados nos extremos da rede, houve um acréscimo de tempo na autenticação, atingindo uma média de 2000 ms. Por fim, o maior resultado alcançado no processo foi a troca de chaves *Diffie-Hellman Efêmero*, com a utilização de criptografia *RSA*.

7. Conclusão

Neste artigo, foi introduzido um modelo de autenticação mútua em ambientes *IoT* com contexto em *Fog Computing* utilizando protocolos já existentes. A autenticação é realizada durante o *handshake* da função "desafio-resposta" (*Fdr*) e através da utilização de chaves assimétricas. A aplicação foi avaliada utilizando ambientes *IoT* reais, mostrando que nossa arquitetura fornece integridade para autenticação através da utilização de função resumo (*Hash*) na assinatura dos parâmetros nas etapas 5 e 6 da Figura 2, como também confidencialidade dos dados trafegados a partir da etapa 5, e, por último, autenticidade com a função "desafio-resposta" com as chaves assimétricas. A contribuição deste trabalho, em relação aos trabalhos anteriores, é a autenticação mútua com simples protocolo de função "desafio-resposta" de mão dupla. Para trabalhos futuros, propõe-se aprimorar o modelo com a utilização de vários tipos de cifras, usando suítes de *handshake*, semelhante ao utilizado no *TLS/SSL* (*Transport Layer Security/Secure Sockets Layer*).

Referências

Atzori, L., Iera, A., and Morabito, G. (2010). The internet of things: A survey. *Computer networks*, 54(15):2787–2805.

Bonomi, F., Milito, R., Zhu, J., and Addepalli, S. (2012). Fog computing and its role in the internet of things. In *Proceedings of the first edition of the MCC workshop on Mobile cloud computing*, pages 13–16. ACM.

Jan, M. A., Nanda, P., He, X., Tan, Z., and Liu, R. P. (2014). A robust authentication scheme for observing resources in the internet of things environment. In *Trust, Security and Privacy in Computing and Communications (TrustCom), 2014 IEEE 13th International Conference on*, pages 205–211. IEEE.

Kothmayr, T., Schmitt, C., Hu, W., Brünig, M., and Carle, G. (2013). Dtls based security and two-way authentication for the internet of things. *Ad Hoc Networks*, 11(8):2710–2723.

Liu, J., Xiao, Y., and Chen, C. P. (2012). Authentication and access control in the internet of things. In *Distributed Computing Systems Workshops (ICDCSW), 2012 32nd International Conference on*, pages 588–592. IEEE.

Rewagad, P. and Pawar, Y. (2013). Use of digital signature with diffie hellman key exchange and aes encryption algorithm to enhance data security in cloud computing. In *Communication Systems and Network Technologies (CSNT), 2013 International Conference on*, pages 437–439. IEEE.

Schneier, B. (1995). *Applied Cryptography (2Nd Ed.): Protocols, Algorithms, and Source Code in C*. John Wiley & Sons, Inc., New York, NY, USA.

Xia, F., Yang, L. T., Wang, L., and Vinel, A. (2012). Internet of things. *International Journal of Communication Systems*, 25(9):1101.

Yi, S., Li, C., and Li, Q. (2015). A survey of fog computing: concepts, applications and issues. In *Proceedings of the 2015 Workshop on Mobile Big Data*, pages 37–42. ACM.