

Mecanismo de Autenticação de Dispositivos para Internet das Coisas

Jonathan Monteiro Araujo , André Peres

¹Instituto Federal de Educação, Ciência e Tecnologia do Rio Grande do Sul (IFRS)
Campus Porto Alegre

jonathanmaraujo@gmail.com, andreperes@poa.ifrs.edu.br

Abstract. *Considering the growth in the internet of things solutions and the number of security vulnerabilities in these solutions, it becomes evident the increased need of security mechanisms for these devices. Implementation failures in one of the main attributes of security, the authentication, is one of the exploited spots that make these devices perform malicious actions. This paper describes the development of an authentication mechanism whose aim is to be used in the devices authentication and communication process in devices, therefore improving their security. The Arduino platform was used in this project to implement and validate the mechanism. As result of this project and the studies carried out it was obtained .a mechanism was developed, composed by: a device register system, an authentication API and a solution able to realize the device authentication in the server.*

Resumo. *Considerando a grande e crescente oferta de soluções para a internet das coisas e o número de falhas de segurança envolvendo tais soluções, fica evidente a necessidade de incremento de mecanismos para a implementação de segurança nesses dispositivos. Falhas na implementação de um dos principais atributos de segurança, a autenticação, é um dos pontos explorados para que estes dispositivos aceitem realizar ações maliciosas. Este trabalho descreve o desenvolvimento de um mecanismo cuja proposta é a de ser utilizado no processo de autenticação e comunicação de dispositivos, oferecendo assim um pouco mais de segurança aos mesmos. No trabalho foi utilizada a plataforma arduino para a implementação e validação do mecanismo. Como resultado deste projeto e dos estudos realizados foi desenvolvido um mecanismo composto por: uma aplicação de cadastro de dispositivos, uma API de autenticação e uma solução capaz de realizar a autenticação do dispositivo no servidor.*

1. Introdução

Atualmente percebe-se que a tecnologia está cada vez mais integrada na vida da população. Um dos aspectos mais notáveis dessa integração pode ser percebido no uso frequente de soluções baseadas em *IoT* (*Internet of Things*). Segundo a *Gartner*, empresa referência no mercado de tecnologia desde 1979, no ano de 2016 o número de dispositivos baseados em *IoT* chegará a 6,4 bilhões, o que corresponde a um aumento de 30% em relação a 2015, com a expectativa de atingir a marca de 20,8 bilhões até 2020 [Gartner 2015]. Desde *smarthomes* (casas inteligentes) até os *wearables* (tecnologias vestíveis), a *IoT* vem crescendo e se integrando ao cotidiano.

No contexto de internet das coisas, uma "coisa" é definida como um objeto capaz de receber dados do ambiente através de sensores, e possivelmente alterar esse ambiente através de atuadores além de comunicar-se com outros objetos [Alecrim 2016]. Os dados recebidos pelo objeto precisam ser processados, sendo esse processamento realizado por um dispositivo controlador conectado à rede.

Com o rápido avanço da oferta de soluções *IoT* e o desejo de difusão dessa tecnologia por parte dos fabricantes, tem-se um cenário onde nota-se pouca preocupação por parte dos desenvolvedores em relação a configurações de soluções de segurança disponíveis para autenticação de dispositivos. Isto resulta em uma possível vulnerabilidade quando leva-se em conta que esses dispositivos podem ser responsáveis pelo gerenciamento de aspectos sensíveis de usuários [Barcena and Wueest 2015].

Este trabalho tem por objetivo o desenvolvimento de um mecanismo de autenticação entre dispositivos na *IoT*. Para o desenvolvimento deste mecanismo será utilizado um dispositivo comumente empregado na prototipação de soluções *IoT*, a plataforma Arduino.

Devido às limitações de processamento e memória desta plataforma, a hipótese deste trabalho parte da seguinte questão: seria possível empregar um mecanismo de autenticação a ser utilizado no desenvolvimento de soluções em *IoT* que utilizam Arduino?

2. Proposta

Visando o aumento da segurança em redes de dispositivos *IoT* e principalmente com o intuito de aplicação de regras de autenticação entre os dispositivos, foi desenvolvido um mecanismo capaz de identificar e autenticar tais dispositivos. Ao ser enviada uma nova transmissão o mecanismo é capaz de verificar em um servidor de autenticação se o dispositivo solicitante está autenticado na rede e, desta forma, apto a receber informações. A solução também deve ser utilizada pelo dispositivo que receberá as informações, pois o mesmo somente deverá ser capaz de receber transmissões de dispositivos autenticados na rede.

O público-alvo deste mecanismo são desenvolvedores que têm como objetivo a construção de soluções de *IoT*, e garantindo com o uso do mecanismo que seus dispositivos somente receberão dados de outros dispositivos autenticados, evitando dessa forma que suas soluções sejam comprometidas e utilizadas com outros fins.

A estrutura de comunicação entre os dispositivos da rede é composta por um servidor e seus dispositivos, o servidor é responsável pelo cadastro, autenticação e comunicação dos dispositivos.

Para o cadastro dos dispositivos foi implementada uma página web, onde é cadastrado o nome e o endereço de rede do dispositivo. Após o envio do formulário com as informações do dispositivo, são gerados: um número identificador do dispositivo, uma senha e uma chave de criptografia. Essas informações são armazenadas em dois locais: primeiramente no banco de dados do servidor e em seguida em um arquivo de texto que é, após a confirmação do cadastro, disponibilizado para download pelo usuário. O arquivo, denominado "auth.txt", precisa ser gravado em um cartão micro SD que por sua vez deve ser inserido no leitor de cartões de uma shield ethernet (placa para comunicação do Arduino), conforme ilustrado na figura 1(A).

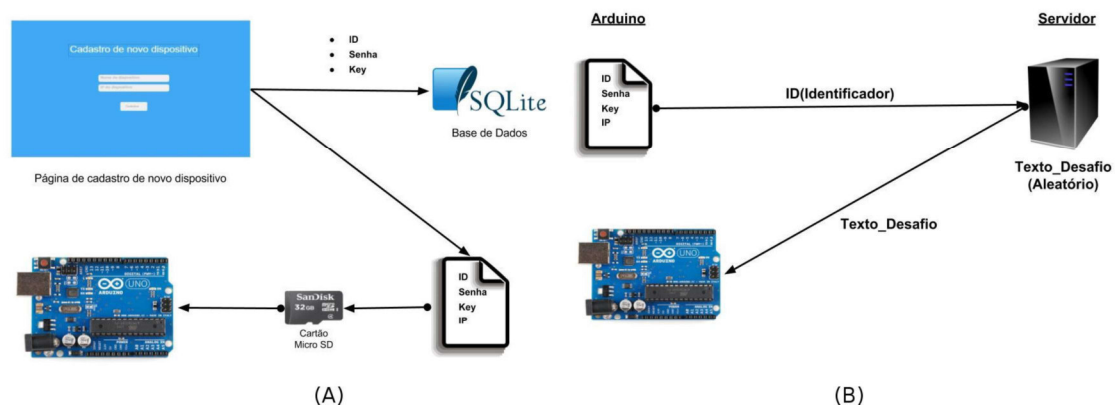


Figura 1. Cadastro de novo dispositivo

Após cadastrado no servidor toda a comunicação entre os dispositivos de uma rede irá passar pelo servidor de autenticação. Para que este possa transmitir informações de um membro da rede para outro de forma segura é indispensável o estabelecimento de um canal de comunicação criptografado. Para o estabelecimento desse canal é preciso que o dispositivo passe pelo seguinte processo de autenticação:

1. O arduino irá ler seu número de identificação no arquivo armazenado no cartão micro SD e enviá-lo para o servidor. Este irá gerar um texto aleatório (texto desafio) e enviá-lo para o arduino, conforme descrito na figura 1(B).
2. O arduino ao receber o texto irá concatená-lo com a senha, que está no arquivo "auth.txt", e calculará o *Hash MD5* do resultado dessa concatenação. Com o *Hash* calculado o dispositivo irá gerar um valor aleatório de 2 bytes denominado "proc". Este é adicionado ao final do *Hash*. O resultado desse processo é então cifrado, utilizando o algoritmo de chave simétrica RC4, com a chave de criptografia gerada no momento do cadastro do dispositivo e armazenada no cartão SD. Em seguida o texto cifrado é enviado para o servidor. Esse processo está demonstrado na figura 2(A).
3. O texto criptografado é decifrado no servidor e uma validação das informações é realizada com base nos registros da base de dados. Caso a validação ocorra com sucesso o dispositivo está autenticado. O servidor então irá enviar uma mensagem de confirmação cifrada contendo o "proc" gerado pelo arduino, seguido de um novo valor aleatório gerado pelo servidor denominado "proc_c", conforme mostrado na figura 2(B).
4. O dispositivo então decifra a mensagem de confirmação e valida o "proc". A partir dessa etapa ambos servidor e dispositivo estão autenticados entre si. A mensagem enviada em seguida contém a informação que deverá ser recebida pelo próximo arduino da rede e deverá conter o "proc_c" do servidor e um novo "proc" gerado pelo arduino. Dessa forma é estabelecido um canal criptografado de comunicação, e a cada nova troca de mensagens, entre o servidor e o arduino, os valores de "proc" e "proc_c" são alterados, conforme demonstrado na figura 2(C).

A implementação dos valores aleatórios "proc" e "proc_c" possuem como principal finalidade a prevenção de ataques do tipo *replay* ou *playback* [Dickson 2016]. Esse tipo de ataque consiste na interceptação e no reenvio de um ou mais pacotes válidos que não

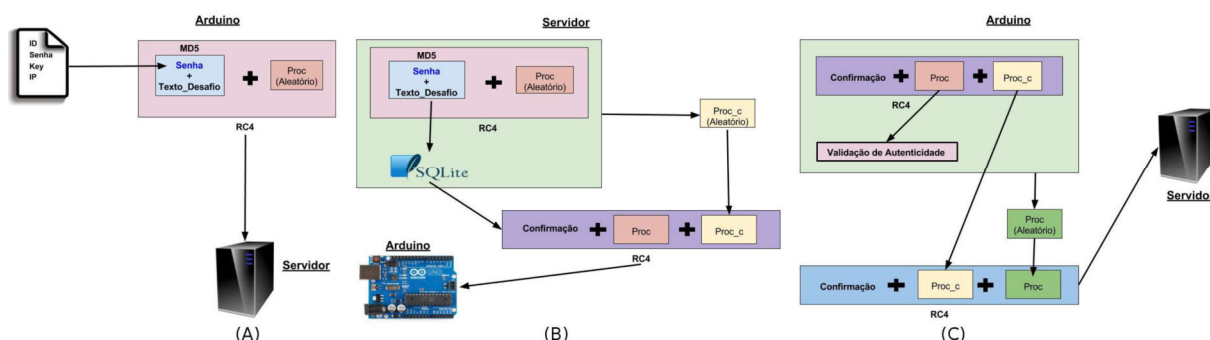


Figura 2. Autenticação

pertencem ao atacante, ataques dessa natureza são muito comuns, haja vista que o atacante nem mesmo precisa ter acesso às chaves de criptografia utilizadas no tráfego.

3. Tecnologias Empregadas e Metodologia

Inicialmente para o desenvolvimento do mecanismo foi utilizado como base o arduino de modelo *UNO*, ou genuíno *UNO*. Esse modelo de arduino havia sido escolhido devido ao fato de ser popular e acessível, considerando a sua disponibilidade e custo. Entretanto devido a limitações de hardware do modelo *UNO* viu-se a necessidade de migração para o arduino do modelo *MEGA 2560*, ou Genuíno *MEGA 2560*.

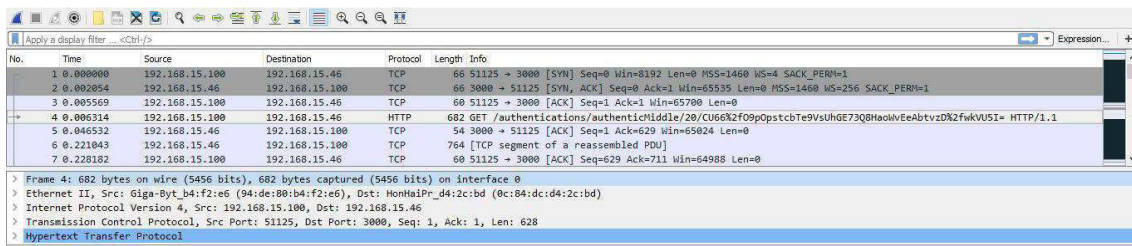
Para o desenvolvimento deste mecanismo foi definida a utilização do Arduino Ethernet Shield V2, que dispõem de uma interface do tipo RJ45 para conexão com a rede através de um cabo padrão. Outra vantagem na utilização deste shield consiste na disponibilização por padrão de uma interface para cartão do tipo micro-SD, o qual é útil no mecanismo para armazenar informações empregadas no processo de autenticação.

Para cifrar e decifrar os textos requisitados na autenticação foi definido o uso do algoritmo RC4. Criado em 1967 por Ronald Rivest o algoritmo RC4 ou ARC4 consiste em um algoritmo de chave simétrica de cifra de fluxo [Stallings 2014], ou seja, o processo de cifragem e decifragem independem do tamanho da chave e as operações são orientadas a byte. O RC4 desempenha a tarefa fundamental de garantir a confidencialidade na transmissão das informações no mecanismo, e demonstrou um bom desempenho ao ser implementado na plataforma arduino, sendo executado rapidamente considerando as limitações da plataforma. Um problema enfrentado com a cifragem utilizando RC4 foi o uso pelo algoritmo de caracteres especiais, esse problema foi tratado através do uso de codificação *Base64* no texto já cifrado.

4. Cenário de Testes e Resultados Obtidos

O cenário de testes foi construído para validar o processo de autenticação e envio de uma mensagem entre dois dispositivos arduino. Neste cenário foram empregados os seguintes ativos: um atacante (executor do teste), dois dispositivos arduino e um servidor de autenticação.

Os testes foram realizados através da execução de um software na rede capaz de analisar o tráfego dos dados, neste teste foi usado o *Wireshark*. A figura 3 apresenta uma



No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	192.168.15.100	192.168.15.46	TCP	66	51125 → 3000 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=4 SACK_PERM=1
2	0.002854	192.168.15.46	192.168.15.100	TCP	66	3000 → 51125 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460 WS=256 SACK_PERM=1
3	0.005569	192.168.15.100	192.168.15.46	TCP	60	51125 → 3000 [ACK] Seq=1 Ack=1 Win=65700 Len=0
4	0.006314	192.168.15.100	192.168.15.46	HTTP	682	GET /authentications/authenticMiddle/20/CU66X2f09p0p0stcbTe9VuhGE73Q@HaoWVeeAbtvzD2fwkVU5I= HTTP/1.1
5	0.046532	192.168.15.46	192.168.15.100	TCP	54	3000 → 51125 [ACK] Seq=1 Ack=629 Win=65824 Len=0
6	0.221843	192.168.15.46	192.168.15.100	TCP	764	[TCP segment of a reassembled PDU]
7	0.228182	192.168.15.100	192.168.15.46	TCP	60	51125 → 3000 [ACK] Seq=629 Ack=711 Win=64988 Len=0

Frame 4: 682 bytes on wire (5456 bits), 682 bytes captured (5456 bits) on interface 0
 Ethernet II, Src: Giga-Byt_b4:f2:e6 (94:de:00:b4:f2:e6), Dst: HontaiPr_d4:2c:bd (0c:84:dc:d4:2c:bd)
 Internet Protocol Version 4, Src: 192.168.15.100, Dst: 192.168.15.46
 Transmission Control Protocol, Src Port: 51125, Dst Port: 3000, Seq: 1, Ack: 1, Len: 628
 Hypertext Transfer Protocol

Figura 3. Exemplo de captura de pacote no software *Wireshark*.

captura de pacote do mecanismo, é possível ver que na requisição obtida é apresentado o texto cifrado assim como o identificador do dispositivo.

Para simular um dispositivo malicioso na rede que tenta se autenticar através do envio das requisições repetidas (ataque de *replay*) para o servidor de autenticação utilizou-se o software *Postman*. A figura 4 demonstra uma tentativa de autenticação que resultou em falha pois foi re-enviada para o *webservice* uma requisição, o erro ocorreu pois o "proc" fornecido já havia sido utilizado anteriormente.



GET `{{address}}{{authenticFinish}}19/oleWmwu4wESLA5MyhOu5fmVQRU=` Params

Authorization Headers Body Pre-request Script Tests

Type No Auth


Body Cookies Headers (10) Tests

Pretty Raw Preview

Failure: Invalid proc number! Authentication failed!

Figura 4. Exemplo de requisição no software *Postman*.

A figura 5 apresenta os logs de monitoramento de um arduino, nele é possível ver os passos de um processo de autenticação bem sucedido.



```

COM6 (Arduino/Genuino Mega or Mega 2560)

Authentication Example v1.0 release 30/06/2017

Local IP: 192.168.15.100

Send an 'g' in serial monitor to start authentication

Authentication Start Initiated
Sending request: GET /authentications/authenticStart/20 HTTP/1.1
Challenge received = b8b10971d2ffef6f
Authentication Start Finished

Authentication Middle Initiated
Sending request: GET /authentications/authenticMiddle/20/X94Q9wFiwhrIat9Xz0+4sRmQ04pUU0w4CJz63QL09f3Daw= HTTP/1.1
Received: tZqRrAKzLf0W0cFZwMwRjRh73WFUlvRAWwTomTg==
Authentication Middle Finished

Authentication Finish Initiated
Message: Turn_On
Sending request: GET /authentications/authenticFinish/20/oIeWmwu4wESLA5MyhOu5fmVQRU= HTTP/1.1
Received Decyphred: Message received
Authentication Finish Finished

Awaiting new orders...
  
```

Figura 5. Processo de autenticação de dispositivo.

O projeto resultou em uma solução capaz de autenticar o envio e o recebimento de informações entre os dispositivos de uma rede, servindo como mediador das transmissões.

O mecanismo foi concebido de forma a ser utilizado por desenvolvedores de soluções IoT, fornecendo assim maior segurança em suas aplicações.

5. Conclusão

O desafio deste trabalho foi o de desenvolver um mecanismo de autenticação capaz de ampliar a segurança de soluções de IoT que utilizam a plataforma Arduino. A capacidade de processamento e a disponibilidade de memória do Arduino acabam por limitar o número de soluções eficazes para este mecanismo.

Considera-se que o mecanismo desenvolvido atinge os objetivos propostos e apresenta como contribuição uma possibilidade de uso e estudo para desenvolvedores de aplicações IoT.

Através dos estudos realizados e apresentados, conclui-se que a IoT é uma área em expansão e que logo a população irá depender do uso dessas soluções. Todavia a dependência dessas tecnologias para a realização de tarefas fundamentais torna a questão de garantia de segurança nestas soluções um assunto relevante e de suma importância, portanto novos estudos são necessários e novas soluções precisam ser desenvolvidas e difundidas.

Durante o desenvolvimento e a execução dos testes do mecanismo alguns pontos de melhoramento do software foram levantados, levando-se em consideração o desempenho, a qualidade do código desenvolvido e a visão de um produto final.

Considerando que este trabalho objetiva primariamente a garantia de autenticidade de dispositivos e todos os atributos componentes da segurança da informação, fica clara a necessidade de novas soluções que contemplem esses atributos. Para tal uma boa solução seria o desenvolvimento de um *framework* visando a disponibilização de métodos e mecanismos implementados com as boas práticas de segurança, dessa forma os desenvolvedores de aplicações IoT poderão, de certa forma, abstrair a segurança e concentrar seus esforços no desenvolvimento da aplicação em si.

Referências

- Alecrim, E. (2016). O que é Internet das Coisas (Internet of Things)? [Online: <http://www.infowester.com/iot.php> Acessado em 18/11/2016].
- Barcena, M. B. and Wueest, C. (2015). Insecurity in the Internet of Things. [Online: <https://www.symantec.com/content/dam/symantec/docs/white-papers/insecurity-in-the-internet-of-things-en.pdf>].
- Dickson, B. (2016). How to Prevent Replay Attacks on Your Website. [Online: <https://www.sitepoint.com/how-to-prevent-replay-attacks-on-your-website/> Acessado em 21/05/2017].
- Gartner (2015). Gartner Says 6.4 Billion Connected "Things" Will Be in Use in 2016, Up 30 Percent From 2015. [Online: <http://www.gartner.com/newsroom/id/3165317> Acessado em 10/11/2016].
- Stallings, W. (2014). *Criptografia e segurança de redes Princípios e práticas*, volume 4. PEARSON Prentice Hall.