INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA

DO RIO GRANDE DO NORTE

NEWTON TEIXEIRA DE BRITO NETO

**GAMESAVED: APLICATIVO MOBILE PARA CATALOGAÇÃO DE JOGOS COM FIREBASE E RAWG**

NATAL - RN

2022

NEWTON TEIXEIRA DE BRITO NETO

**GAMESAVED: APLICATIVO MOBILE PARA CATALOGAÇÃO DE JOGOS COM FIREBASE E RAWG**

Trabalho de Conclusão de Curso apresentado ao Curso Superior de Tecnologia em Redes de Computadores do Instituto Federal de Educação, Ciência e Tecnologia do Rio Grande do Norte, em cumprimento às exigências legais como requisito parcial à obtenção do título de Tecnólogo em Redes de Computadores.

Orientador: NOME DO PROFESSOR ORIENTADOR

NATAL - RN

2017

Brito Neto, Newton Teixeira de.

B862a Active unlock: proposta de controle de acesso integrado ao active directory utilizando NFC e tecnologias open-source / Newton Teixeira de Brito Neto. – Natal, 2018.

59 f : il. color.

Trabalho de Conclusão de Curso (Tecnologia em Redes de Computadores) – Instituto Federal de Educação, Ciência e Tecnologia do Rio Grande do Norte. Natal, 2018.

Orientador (a): M.e Thiago Fernandes Silva Dutra.

1. Sistema de segurança. 2. Active Directory. 3. Near Field Communication. 4. Arduino. 5. PHP (Linguagem de programação de computador). I. Dutra, Thiago Fernandes Silva. II. Instituto Federal de Educação, Ciência e Tecnologia do Rio Grande do Norte. III. Título.

CDU 004.7

Catalogação na Publicação elaborada pela Bibliotecária Roberta Jerônimo da Silva CRB15: 761

Biblioteca Central Sebastião Fernandes (BCSF) - IFRN

NEWTON TEIXEIRA DE BRITO NETO

**GAMESAVED: APLICATIVO MOBILE PARA CATALOGAÇÃO DE JOGOS COM FIREBASE E RAWG**

Trabalho de Conclusão de Curso apresentado ao Curso Superior de Tecnologia em Redes de Computadores do Instituto Federal de Educação, Ciência e Tecnologia do Rio Grande do Norte, em cumprimento às exigências legais como requisito parcial à obtenção do título de Tecnólogo em Redes de Computadores.

Trabalho de Conclusão de Curso apresentado e aprovado em \_\_\_ /\_\_\_/\_\_\_\_, pela seguinte Banca Examinadora:

BANCA EXAMINADORA

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

NOME – Presidente

Instituto Federal de Educação, Ciência e Tecnologia do Rio Grande do Norte

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

NOME – Examinador

Instituto Federal de Educação, Ciência e Tecnologia do Rio Grande do Norte

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

NOMAE – Examinador

Instituto Federal de Educação, Ciência e Tecnologia do Rio Grande do Norte

Dedico este trabalho a minha família, amigos e colegas de trabalho que me ajudaram e guiaram durante toda a minha jornada acadêmica.

**AGRADECIMENTOS**

Agradeço primeiramente a minha mãe, Keila Roxana Pereira por ter paciência quanto a conclusão deste trabalho.

Agradeço também a todos os meus familiares, principalmente ao meu pai Ednaldo Jacinto de Brito, minha vó Santina Maria de Oliveira e a minha tia Nadir, por todo o apoio prestado nessa jornada.

Aos meus chefes do trabalho, Carolina Barbosa e Afonso Leirias, que me permitiram desenvolver o trabalho durante os momentos vagos, e também ao colega Hélio Feliciano por sanar várias das minhas dúvidas referentes a algumas tecnologias utilizadas neste trabalho.

Aos amigos que fiz durante o período do curso, Diego Filgueira, Jonatha Nogueira, Joewerton César, Marcelo Ciríaco e William Furtado, esses que fizeram a vida acadêmica ser mais interessante e divertida, amigos que espero levar durante a vida toda.

Sou muito grato também a todos os professores que tive o prazer de conhecer durante a minha estadia no curso, eles que me ajudaram na busca pelo conhecimento e capacitação profissional, e por fim ao meu orientador, Thiago Fernandes Silva Dutra, que acreditou no meu projeto e me deu o auxílio necessário para desenvolver o mesmo.

**RESUMO**

O seguinte trabalho apresenta os detalhes do desenvolvimento de um projeto que visa entregar um aplicativo mobile para entusiastas de jogos digitais salvarem lista de jogos e interagirem em comunidades para jogadores. Para viabilizar o citado projeto, é utilizado o serviço *Firebase* do Google, que fornece os serviços de autenticação, base de dados e *storage* de arquivos, e a *Application Programming Interface* (API) externa RAWG, que disponibiliza as informações sobre os jogos. O GameSaved permite ao seu usuário buscar por diversos jogo de diversas plataformas, obtendo assim informações sobre o mesmo e dando a possibilidade de salvá-lo em uma lista, sendo possível também buscar por outros usuários da aplicação e visualização de alguns dados sobre o mesmo, além de poder interagir com outros usuários através de fóruns.

Palavras-Chave: Firebase. RAWG. GameSaved. Jogo. Fóruns.

**ABSTRACT**

The following work presents details about a project that aims to facilitate access to controlled environments, integrating an Active Directory (AD) user base through the use of Near Field Communication (NFC) and open-source technologies. The project consists of an Arduino prototype board connected to a tag reader module based on NFC technology, which can read the tags and will capture user stored data and send it to a web service in Hypertext Preprocessor (PHP). This service in particular will verify if the received credentials are registered to a directory service Lightweight Directory Acess Protocol (LDAP) called Active Directory. In case there is a web service, a message will be returned to Arduino indicating that the user exists, releasing the access to the service solicited by the user, otherwise , in case that the response received about the verified data of the Arduino are nonexistent, the user access will be denied. A prototype device was also developed with the ability to insert information into the memory of the tags using a selected memory block, as the information entered refers to the User and password registered in Active Directory.

Keywords: Active Directory. LDAP. Arduino. NFC. PHP.

**LISTA DE ILUSTRAÇÕES**

[Quadro 1 - Modos suportados por cada tipo de dispositivo. 17](#_Toc508548209)

[Figura 1 - Exemplo de uso do NFC 18](#_Toc508548210)

[Figura 2 - Trocando dados com emparelhamento NFC 19](#_Toc508548211)

[Figura 3 - Usando o NFC para pagamentos com cartão de crédito virtual 20](#_Toc508548212)

[Figura 4 - Representação de uma das várias formas de *smart card* 20](#_Toc508548213)

[Figura 5 - Exemplo de topologia de Unidades Organizacionais 23](#_Toc508548214)

[Figura 6 - Exemplo de hierarquia utilizando UO 24](#_Toc508548215)

[Figura 7 - Exemplo de topologia de distribuição de políticas 25](#_Toc508548216)

[Figura 8 - Arduino Uno, atualmente a versão de Arduino mais popular 28](#_Toc508548217)

[Figura 9 - Arduino Uno integrado a um *shield ethernet* 28](#_Toc508548218)

[Figura 10 - Shield ethernet equipado com o chip Wiznet W5100 29](#_Toc508548219)

[Figura 11 - Modulo PN532 30](#_Toc508548220)

[Figura 12 - Modulo MFRC522 30](#_Toc508548221)

[Figura 13 - Funcionamento do sistema 33](#_Toc508548222)

[Figura 14 - SET0 e SET1 34](#_Toc508548223)

[Figura 15 - Protótipo de leitura de tags 35](#_Toc508548224)

[Figura 16 - Protótipo utilizado para gravar usuário e senha nos cartões 36](#_Toc508548225)

[Figura 17 - Dados armazenados em uma tag 37](#_Toc508548226)

[Figura 18 - Resultado do cenário 1 40](#_Toc508548227)

[Figura 19 - Resultado do cenário 2 41](#_Toc508548228)

[Figura 20 - Resultado do cenário 3 42](#_Toc508548229)

[Figura 21 - Resultado do cenário 4 43](#_Toc508548230)

[Figura 22 - Resultado do cenário 5 44](#_Toc508548231)

[Figura 23 - Resultado do cenário 6 45](#_Toc508548232)

**LISTA DE ABREVIATURAS E SIGLAS**

API Application Programming Interface

AD Active Directory

DAP Directory Access Protocol

GPO Group Policy Object

HTML HyperText Markup Language

IDE Integrated Development Environment

IEC International Electrotechnical Commission

IETF Internet Engineering Task Force

IMAP Internet Message Access Protocol

ISO International Organization for Standardization

kbps kilobit per second

LDAP Lightweight Directory Access Protocol

LED Light Emitting Diode

MHz Megahertz

MIFARE Mikron FARE Collection System

NFC Near Field Communication

NFCIP-1 Near Field Communication Interface and Protocol 1

OSI Open System Interconnection

PHP Hypertext Preprocessor

PICC Proximity Integrated Circuit Card

POE Power Over Ethernet

POP3 Post Office Protocol

RFID Radio Frequency Identification

SMS Short Message Service

SNMP Simple Network Management Protocol

SQL Structured Query Language

TCP Transmission Control Protocol

UDP User Datagram Protocol

UO Unidade Organizacional

URL Uniform Resource Locator

USB Universal Serial Bus

Wi-Fi Wireless Fidelity

**SUMÁRIO**

[**1 INTRODUÇÃO** 12](#_Toc492922842)

[1.1 MOTIVAÇÃO 12](#_Toc492922843)

[1.2 OBJETIVO GERAL 13](#_Toc492922844)

[1.3 OBJETIVOS ESPECÍFICOS 13](#_Toc492922845)

[1.4 ESTRUTURA DO TRABALHO 14](#_Toc492922846)

[**2 MATERIAIS E MÉTODOS** 15](#_Toc492922847)

[2.1 NEAR FIELD COMMUNICATION 15](#_Toc492922848)

[2.1.1 **Comportamento Ativo e Passivo** 17](#_Toc492922849)

[2.1.2 **Initiator e Target** 17](#_Toc492922850)

[2.1.3 **Modos de Operação** 18](#_Toc492922851)

[2.1.3.1 Modo Leitor/Escritor 18](#_Toc492922852)

[2.1.3.2 Peer-to-Peer 19](#_Toc492922853)

[2.1.3.3 PICC 19](#_Toc492922854)

[2.1.4 **Smart Cards NFC** 20](#_Toc492922855)

[2.1.4.1 MIFARE 21](#_Toc492922856)

[2.2 WINDOWS SERVER 2012 21](#_Toc492922857)

[2.3 ACTIVE DIRECTORY 22](#_Toc492922858)

[2.3.1 **Unidades Organizacionais** 23](#_Toc492922859)

[2.3.2 **Group Policy Object** 24](#_Toc492922860)

[2.4 HYPERTEXT PREPROCESSOR (PHP) 26](#_Toc492922861)

[2.5 ARDUINO 27](#_Toc492922862)

[2.5.1 **Shield Ethernet W5100** 28](#_Toc492922863)

[2.5.2 **Módulo NFC PN532** 29](#_Toc492922864)

[2.5.3 **Módulo RFID MFRC522** 30](#_Toc492922865)

[**3 DESENVOVIMENTO DO SISTEMA** 31](#_Toc492922866)

[3.1 PRÉ-REQUISITOS E ARQUITETURA DE DESENVOLVIMENTO 31](#_Toc492922867)

[3.2 COLETA DE DADOS 32](#_Toc492922868)

[3.3 MONTAGEM DO SISTEMA 33](#_Toc492922869)

[**4 TESTES E RESULTADOS** 39](#_Toc492922870)

[4.1 CENÁRIO 1 39](#_Toc492922871)

[4.2 CENÁRIO 2 40](#_Toc492922872)

[4.3 CENÁRIO 3 42](#_Toc492922873)

[4.4 CENÁRIO 4 43](#_Toc492922874)

[4.5 CENÁRIO 5 43](#_Toc492922875)

[4.6 CENÁRIO 6 44](#_Toc492922876)

[**5 CONCLUSÃO E TRABALHOS FUTUROS** 46](#_Toc492922877)

[**REFERÊNCIAS** 48](#_Toc492922878)

[**APÊNDICE A** – Código fonte do sistema 51](#_Toc492922879)

[**APÊNDICE B** – Código fonte do sistema 56](#_Toc492922880)

# INTRODUÇÃO

No mundo globalizado e atento aos avanços da tecnologia da informação que vivemos atualmente, temos uma busca cada vez maior pela adoção de novas tecnologias em nosso dia a dia. Uma prova dessa adoção é a crescente demanda pelo uso de novas tecnologias para liberação de acesso à alguma área com a intenção de se libertar do antigo uso das chaves que podem ser facilmente perdidas e/ou clonadas. Soluções desse tipo já são encontradas atualmente em diversos hotéis, através do uso de tecnologias baseadas em rádio frequência, agradando os seus clientes com uma abordagem mais moderna. Dentre as tecnologias atuais de rádio frequência, uma das que se mostra mais promissora é a *Near Field Communication* (Comunicação por Campo de Proximidade), largamente conhecida pela sigla NFC.

Outra tendência ligada à tecnologia da informação nos dias atuais, é a centralização da base de dados de autenticação e autorização. Para tal, costuma-se utilizar um sistema baseado em *Lightweight Directory Access Protocol* (Protocolo de Acesso aos Diretórios Leves), LDAP, para gerenciar e acessar informações de diretório. Um desses sistemas que se encontra amplamente difundido é o *software* *Microsoft Active Directory*, encontrado nos sistemas operacionais *Windows Server* (a partir da versão 2000) para servidores.

Esse trabalho visou desenvolver um projeto integrando essas duas tecnologias, o *Active Directory* e NFC, com o intuito de realizar o controle de acesso de usuários. Para validar essa proposta, um protótipo de fechadura eletrônica foi construído utilizando-se uma placa de prototipagem Arduino para intermediar a comunicação entre um dispositivo NFC e o Active Directory para liberar ou não o acesso de uma pessoa a uma determinada área.

Os resultados mostraram que esse trabalho fornece uma ferramenta de controle de acesso acessível baseada em NFC, com seu *software* sendo disponibilizado de forma *open source* e seu *hardware* podendo ser facilmente construído seguindo-se as especificações contidas neste documento, podendo ser integrada numa rede que já faça uso do *Windows Server* e do *Active Directory*.

## 1.1 MOTIVAÇÃO

Visando uma modernização tecnológica das instituições, foi desenvolvido este trabalho que tem a intenção de apresentar uma solução viável e prática para substituir o arcaico controle de acesso que fazem uso das fechaduras convencionais, que por sua vez se utilizam de chaves pouco cômodas e inseguras, podendo ser facilmente copiadas, além de terem uma reposição inconveniente em caso de perda ou defeito, necessitando de um trabalho manual especializado de um chaveiro para solucionar o problema.

Tomando como exemplo o ambiente do Instituto Federal do Rio Grande do Norte (IFRN), local onde esse trabalho foi desenvolvido, quando os alunos necessitam fazer uso dos laboratórios disponíveis para estudo e pesquisa, os mesmos devem se dirigir à diretoria do curso ao qual são vinculados e solicitar a chave do local desejado. Caso ela esteja disponível, o aluno assina seu nome em uma folha de papel, juntamente com sua matrícula e a hora em que pegou a chave. Um procedimento semelhante também é utilizado pelos professores, que precisam ir até a diretoria responsável para pegar a chave de interesse.

Além da constante necessidade de deslocamento às diretorias, outro grande problema desse processo é que ele pode ser fraudado facilmente. É possível que nas diretorias não exista um funcionário responsável por fiscalizar se o aluno que solicita a chave está preenchendo os dados corretamente. A situação se agrava quando outro aluno, não sendo o mesmo que pegou a chave inicialmente (as vezes acontece do aluno que pegou a chave sair do laboratório e deixa-la com outro aluno), fica responsável por devolver a chave, tal procedimento dificulta a atribuição de responsabilidade caso ocorra algum dano ao ambiente ou equipamentos, esse cenário motivou o desenvolvimento do protótipo.

## 1.2 OBJETIVO GERAL

A proposta principal do projeto foi criar um protótipo tecnológico que possa substituir as atuais formas de controle de acesso que utiliza fechaduras e chaves convencionais. Basicamente, o seu objetivo é fazer um controle de acesso utilizando uma base de usuários já existente na instituição, oferecendo mais segurança e comodidade aos seus usuários. Com esse sistema, o detentor de uma *tag* NFC poderá ter acesso à locais com fechadura eletrônica de uma forma cômoda e auditável.

## 1.3 OBJETIVOS ESPECÍFICOS

1. entregar a instituição dados precisos sobre quem está utilizando os espaços, e quando;
2. implementar um protótipo com capacidade de controle de acesso, com níveis de acesso e registro de uso;
3. desenvolver uma ferramenta de fácil integração a plataforma *Active Directory* sem que nenhuma alteração na mesma seja necessária, e que seja possível utilizar a base de usuários já cadastrada;
4. substituir a utilização das chaves convencionais por etiquetas NFC;
5. criar um protótipo de baixo custo utilizando *software* e *hardware open source*.

## 1.4 ESTRUTURA DO TRABALHO

O trabalho foi dividido em cinco capítulos: Introdução, Materiais e Métodos, Desenvolvimento do Protótipo, Testes e Resultados e Conclusão e Trabalhos Futuros.

No primeiro capítulo temos a introdução, onde é exposto os conceitos inicias do trabalho, além da ideia que motiva o mesmo, por fim, são listados os objetivos desejados.

O segundo capítulo trata dos materiais e métodos utilizados para tornar o projeto uma realidade, abordando cada uma delas de maneira mais profunda, além de trazer uma contextualização histórica das mesmas.

O terceiro capítulo, intitulado de Desenvolvimento do Protótipo, descreve o processo de montagem do protótipo, juntamente com os pré-requisitos e arquitetura de desenvolvimento. É usado também para descrever algumas dificuldades e problemas encontrados no caminho, bem como os meios utilizados para soluciona-los.

Em Testes e Resultados são propostas possíveis situações nas quais a solução seria submetida num ambiente real. Com essas situações em mente, foram realizados diversos testes para validar o correto funcionamento do *Active Unlock* nas mais diversas situações.

No quinto e último capítulo, Conclusão e Trabalhos Futuros, é apresentado o resultado final do projeto e o que foi usado para concluir o protótipo. Neste capítulo são também comentadas possíveis melhorias no projeto atual e funcionalidades que poderiam ser implementadas para aperfeiçoá-lo.

# MATERIAIS E MÉTODOS

Nos tópicos abaixo serão abordados temas referentes as tecnologias que foram utilizadas neste trabalho, assim como uma visão clara sobre os seus funcionamentos.

## 2.1 NEAR FIELD COMMUNICATION

O *Near Field Communication* (Comunicação por Campo de Proximidade, em tradução livre), mais conhecido como NFC, é uma tecnologia *open source* de comunicação sem fio de curto alcance baseado em radiofrequência que foi desenvolvido em uma parceria entre *Sony* e *Philips* em 2002 (NXP SEMICONDUCTORS, 2016).

O uso da tecnologia NFC hoje em dia encontra-se muito difundida, sendo utilizada principalmente nos telefones celulares mais modernos, os *smartphones*, na bilhetagem eletrônica e no controle de acesso. Suas principais aplicações estão nas áreas de:

1. pagamentos: Basicamente, pode-se fazer pagamentos usando um celular equipado com NFC, dispensando assim, o uso dos cartões de crédito convencionais. No Brasil essa tecnologia ainda é pouco difundida, recebendo suporte apenas por parte da *Samsung* com o *Samsung Pay*, presente nos *smartphones* mais modernos da marca (G1, 2016), porém em alguns países como Estados Unidos, Reino Unido, Canadá e Austrália (NFC.org), essa tecnologia já é uma opção mais difundida, contando também com os serviços do *Android Pay* (Google Blog, 2016) e *Apple Pay* (Apple, 2017);
2. troca de dados: O NFC suporta o envio e recebimento de dados digitais, desde que os dois dispositivos envolvidos tenham suporte a tecnologia (NFC FORUM, 2017). Nos *smartphones* é bastante comum o uso para envio de fotos e músicas;
3. chaves eletrônicas: Outro uso para o NFC é o de autenticação para controle de acesso. Já é uma prática adotada em alguns hotéis pelo mundo o uso de cartões equipados com etiquetas NFC para dar acesso aos quartos dos hóspedes (NXP Blog, 2015). A aplicação pode ser expandida para além dos quartos caso seja de interesse da administração do estabelecimento, podendo ser implementada também para controle de acesso em refeitórios, salas de ginástica, lavanderias e áreas restritas apenas para funcionários. A tecnologia pode ser até expandida, utilizando as *tags* NFC embutidas em smartphones ao invés dos *smart cards* convencionais (NFC World, 2011);
4. identificação: Uma técnica pouco comum ainda, mas já conhecida, é implantar etiquetas NFC no próprio corpo, para identificação do portador, desbloquear *smartphone*, desarmar sistemas de alarme e etc. (Redbull, 2016). Por enquanto apenas pequenos chips são implantados, um dos mais adotados é o NFC, devido a sua vasta gama de aplicações e de não precisar de uma fonte constante de alimentação, a mesma acontece durante o contato com um dispositivo ativo;
5. definição de configurações em dispositivos eletrônicos: podemos armazenar, em etiquetas NFC, informações referentes a configurações de dispositivos com suporte ao NFC, e assim que o contato entre os dois for feito, essas configurações são carregadas no dispositivo em questão, poupando o usuário de todo o trabalho manual de configurar o dispositivo. Um exemplo interessante seria colocar as informações necessárias para se conectar a uma rede Wi-Fi (*Wireless Fidelity*) em uma *tag* NFC, então, para que um *smartphone* se conecte a essa rede seria necessário apenas passá-lo sobre a *tag*.

As definições dos esquemas de modulação, codificações de *bit*, taxas de transmissão, formato de quadro e mecanismos de inicialização e controle de colisão do NFC, estão definidos nos documentos ISO/IEC (*International Organization for Standardization*/*International Electrotechnical Commission*) 18092:2013 como NFCIP-1 (*Near Field Communication Interface and Protocol 1*). Segundo essa norma, a banda de frequência para operação de comunicações NFC é 13,56MHz (*Megahertz*), onde também se faz possível a comunicação com dispositivos RFID (*Radio Frequency Identification*) que trabalhem na mesma banda de frequência. Quanto a velocidade máxima da conexão, em uma troca de dados pode chegar até 424kbps (*kilobit per second*, ou quilobit por segundo) em distâncias menores que 4 centímetros (NFC FORUM, 2016).

A comunicação NFC funciona com apenas dois dispositivos simultâneos, um *initiator* (iniciador) e um alvo *target* (alvo). O iniciador, começa e controla o processo de troca de informações. O alvo apenas transmite a informação, ou recebe uma nova. O *initiator* é sempre um dispositivo ativo, além de transmitir informações ele é o responsável por controlar o fluxo de transmissão de dados. Já o *target* pode ser tanto um dispositivo ativo quanto um passivo, que é o caso das *tags* NFC que apenas recebem e transmitem informações passadas ou solicitadas pelo *initiator*. No caso do *target* ser um dispositivo com capacidade ativa, o mesmo vai apenas operar como passivo durante essa comunicação, um exemplo disso é quando dois *smartphones* realizam uma troca de dados via NFC, ambos tem capacidade ativa, mas durante o processo apenas um deles vai assumir o papel passivo.

### 2.1.1 Comportamento Ativo e Passivo

Os dispositivos NFC são classificados em duas categorias diferentes, ativos e passivos, o que os diferencia é como cada um é energizado. Dispositivos ativos são energizados fazendo uso de alguma fonte de energia, como baterias e fontes de alimentação, enquanto os dispositivos passivos não dispõem de nenhum tipo de fonte de energia, a sua energização é feita através do campo magnético gerado pelo dispositivo ativo, ou seja, o dispositivo passivo só é energizado durante uma troca de informações com um dispositivo ativo (COSKUN, OK, OZDENIZCI, 2012).

### 2.1.2 Initiator e Target

A comunicação NFC acontece somente entre dois dispositivos, onde um deles é o *initiator* e o outro é o *target*. O iniciador é quem dá início a comunicação entre os dispositivos, enquanto o alvo é quem vai responder as solicitações do iniciador.

Como o iniciador é quem inicia a comunicação NFC supõe-se que o mesmo seja o dispositivo ativo, já que para iniciar a comunicação é necessário energia, e como citado anteriormente, somente o ativo tem um fonte de energia, pra ser um iniciador o dispositivo obrigatoriamente deverá ser um ativo. No entanto, o alvo pode ser representado por ambos os tipos de dispositivos, tanto ativo quanto passivo. Porém, caso seja um ativo o mesmo usará sua própria fonte de energia (COSKUN; OK; OZDENIZCI, 2012). O Quadro 1 abaixo, ilustra de forma clara que papel (iniciador ou alvo) cada tipo de dispositivo (ativo ou passivo) pode assumir, além de associar o qual possui alimentação constante e qual não possui.

Quadro 1 - Modos suportados por cada tipo de dispositivo.

|  |  |  |  |
| --- | --- | --- | --- |
| Dispositivo | Iniciador | Alvo | Fonte de energia |
| Ativo | Sim | Sim | Possui |
| Passivo | Não | Sim | Não possui |

Fonte: Autoria própria (2017).

### 2.1.3 Modos de Operação

O NFC trabalha com 3 modos de operações diferentes, são elas: Leitor/Escritor, o *Peer-to-Peer* e o *Proximity Integrated Circuit Card* (FERRAZ FILHO, 2010). Ambos serão melhores descritos a seguir.

#### 2.1.3.1 Modo Leitor/Escritor

No modo Leitura/Escrita o dispositivo ativo inicia uma operação ao detectar um dispositivo passivo. A ação tomada pelo dispositivo ativo vai depender do tipo de informação obtida durante a comunicação. Nesse modo, o iniciador inicia uma sessão de comunicação com uma *tag* (alvo passivo), onde o ativo (o iniciador) vai ler as suas informações. Um exemplo desse tipo de conexão é quando um dispositivo equipado com um sensor leitor NFC verifica informações contidas em uma *tag* compatível embutida em algum produto, essa informação pode conter o endereço de um *website* hospedado na rede mundial de computadores que será aberto automaticamente no dispositivo do usuário, além de enviar uma *Short Message Service* (SMS), ou serviço de mensagens curtas, obter detalhes sobre filmes, produtos alimentícios e etc. (STMICROELECTRONICS, 2016). Na Figura 1 podemos ver um pedestre fazendo uso da tecnologia NFC para obter informações sobre um evento.

Figura 1 - Exemplo de uso do NFC



Fonte: RFID INSIDER (2013).

#### 2.1.3.2 Peer-to-Peer

O modo *Peer-to-Peer* acontece quando dois dispositivos ativos estabelecem uma conexão para envio e recebimento de dados. Nesse caso um deles vai agir efetivamente como ativo, sendo o iniciador, enquanto o outro, apesar de ter a mesma capacidade, vai trabalhar como alvo. Um exemplo é quando dois *smartphones* são pareados para compartilhar um ou mais arquivos digitais (músicas, fotos, etc.). A Figura 2 retrata o emparelhamento de dois *smartphones* para compartilhamento de arquivos via NFC.

Figura 2 - Trocando dados com emparelhamento NFC



Fonte: TECHTUDO (2013).

#### 2.1.3.3 PICC

*Proximity Integrated Circuit Card*, ou PICC, basicamente é um emulador de *tag*. Nesse modo, o dispositivo ativo vai parar de gerar seu próprio campo magnético, fazendo o mesmo trabalhar como se fosse apenas uma *tag*, ou seja, como um dispositivo passivo. Essa tecnologia está atualmente em ampla expansão, principalmente nos *smartphones*, onde alguns fabricantes, como *Apple* e *Samsung*, estão usando esse modo de operação para substituir o cartão de crédito dos seus consumidores pelo próprio *smartphone*. Sendo assim, alguém que use essa funcionalidade não precisará mais portar o seu cartão de crédito real, pois o seu *smartphone* já fará a função do mesmo. A Figura 3 retrata o pagamento de uma compra usando cartão de crédito virtual, que usa a tecnologia NFC.

Figura 3 - Usando o NFC para pagamentos com cartão de crédito virtual



Fonte: NFC WORLD (2013).

### 2.1.4 Smart Cards NFC

*Smart card* é como são chamados os pequenos *chip*s encontrados hoje em dia em cartões de crédito/débito, *chips* de celular, certificados digitais e outros. A Figura 4 é uma representação genérica de um *smart card*, onde o mesmo é mostrado como a parte amarela (nem sempre ficando visível para o usuário), a área branca serve apenas para facilitar o uso.

Figura 4 - Representação de uma das várias formas de *smart card*



Fonte: NEWSHAWK TIME (2017)

Ele não inclui periféricos como teclado, monitor, e etc., mas possui todos os elementos da plataforma de computação de propósito geral (JURGENSEN; GUTHERY, 2002). No caso dos *smart cards*, sua arquitetura é formada por uma unidade de processamento, memória RAM, memória ROM e memória EEPROM (UFRJ):

1. unidade de Processamento: processador do tipo *Reduced Instruction Set Computer* (RISC), ou Computador com Conjunto de Instruções Reduzidas, em tradução livre. Com arquitetura variando entre 8, 16 e 32bits e podendo chegar até 35MHz de *clock*;
2. RAM: com uma capacidade média de 512bytes, a RAM é utilizada para armazenar os dados voláteis que vão ser solicitados pelo processador;
3. ROM: memória usada para armazenar dados que não podem ser perdidos, essas informações são gravadas durante a fabricação do chip, como o sistema operacional, usado para controlar o processador. Sua capacidade costuma ser de 16kB;
4. EEPROM: utilizada para armazenar dados de segurança. Ela pode ser apagada por processos elétricos e rescrita até 10.000 vezes. Capacidade de até 64 kB (UNICAMP).

#### 2.1.4.1 MIFARE

MIFARE (*Mikron FARE Collection System*) é a linha de *smart cards,* compatíveis com NFCda *NXP Semiconductors* (antiga *Philips Semiconductors*). Segundo a fabricante, é o modelo de *smart card* mais popular do mundo, sendo compatível com mais de 40 aplicações, além de ter vendido 260 milhões de leitores, e 10 bilhões de cartões (NXP SEMICONDUCTORS, 2016). A linha MIFARE é baseada na ISO 14443 e atualmente conta com 6 modelos, o *Classic*, *DESFire*, *Plus*, *Ultralight* e *SAMs for Reader Systems*.

Os cartões MIFARE são capazes de receber e enviar dados a 106 kbps sem necessariamente realizar um contato físico com um dispositivo ativo, suportando uma distância de até 10 cm dependendo da antena do ativo (NXP SEMICONDUCTORS, 2014). Outra característica importante é a sua independência em relação a uma fonte constante de energia, os cartões MIFARE só precisam ser energizados durante a troca de informações com um dispositivo leitor/escritor (ou seja, um dispositivo ativo), no caso, sua energização acontece via indução eletromagnética (NXP SEMICONDUCTORS, 2014).

## 2.2 WINDOWS SERVER 2012

O *Windows Server 2000* foi lançado no início dos anos 2000 para substituir o *Windows NT Server 4.0*, sendo o quinto sistema operacional da *Microsoft* voltado para servidores. A nova versão do sistema operacional buscava oferecer ao usuário uma maior confiabilidade, disponibilidade e escalabilidade, reduzindo custos através de um gerenciamento simplificado, além de fornecer um servidor robusto para aplicações (WALSHAW, 2000).

A versão mais atual é o *Windows Server 2012*, sendo um sistema operacional robusto com foco em servidores, substituindo sua versão anterior, o *Windows Server 2008*.

Dentre as novidades em relação a versão anterior, possui suporte para computação na nuvem e a interface similar ao *Windows 8* (versão do sistema operacional para uso doméstico e empresarial, com foco no usuário).

Foi lançado no mercado em 4 versões distintas, são elas: *Datacenter*, *Standart*, *Essentials* e *Foundation*.

Um dos principais diferenciais do *Windows Server* quando comparado a outros sistemas operacionais para servidores, é que ele vem equipado com um serviço de diretório, o *Active Directory*, que é baseado no difundido protocolo LDAP.

## 2.3 ACTIVE DIRECTORY

Em 1980, com a expansão da *Internet* e a crescente preocupação com a organização dos diretórios de rede, foi publicada a ISO 9594, ou *Directory Access Protocol* (DAP). O DAP não foi muito utilizado devido ao consumo excessivo de banda e dependência do protocolo *Open System Interconnection* (OSI), que já estava ultrapassado.

Segundo Marco Aurélio (2013, p. 235), devido aos problemas do DAP o *Internet Engineering Task Force*, ou IETF, criou a *Request for Comments* 1777 (RFC 1777), batizada de *Lighweight Directory Access Protocol*, ou LDAP. Com o uso da pilha de protocolo TCP/IP (uma evolução do modelo OSI) e menor consumo de banda, o LDAP conseguiu obter o sucesso que o DAP não teve, sendo utilizado em larga escala.

O *Active Directory*, vastamente conhecido como AD, é o serviço de diretório da *Microsoft* baseado no protocolo LDAP. Está presente na plataforma *Windows Server* desde a versão 2000, sendo seu recurso mais atrativo.

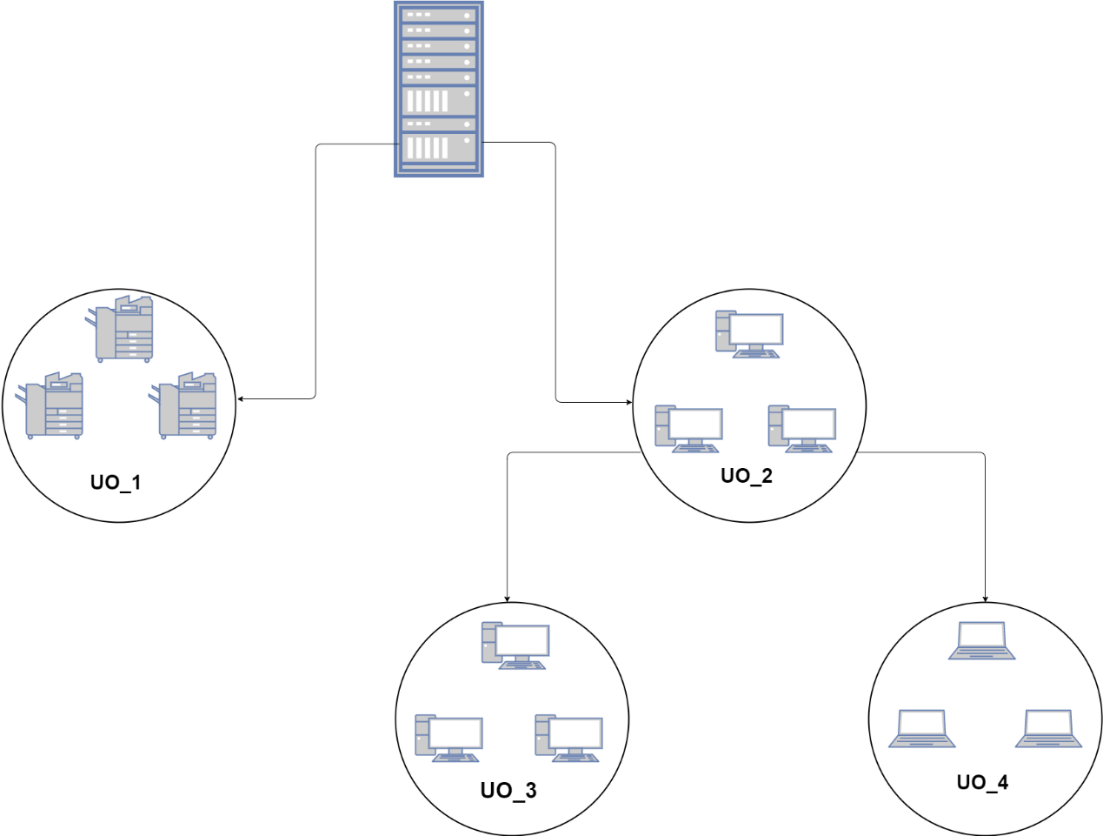
Segundo Stanek (2009), o *Active Directory* é um serviço de diretório capaz de gerenciar os recursos da rede de forma exemplar. O ADarmazena e gerencia informações detalhadas sobre cada objeto da rede, desde usuários e computadores até impressoras, facilitando a pesquisa e autenticação. O foco dos serviços de diretórios é manter sobre controle e de forma organizada, uma grande quantidade de informações, para pesquisa e recuperação das mesmas.

O AD conta com duas ferramentas primordiais que ajudam os administradores de redes a organizarem e controlarem os objetos de uma rede, são elas: as Unidades Organizacionais e as Políticas de Grupo.

### 2.3.1 Unidades Organizacionais

Unidades organizacionais são contêineres criados dentro de um domínio (uma partição de uma floresta Active Directory) e podem conter computadores, perfis de usuários, grupos, contatos, impressoras, pastas compartilhadas e até mesmo outras UOs. Estas unidades organizacionais existem com a finalidade de representar uma hierarquia, em um ambiente coorporativo, por exemplo, poderia ser criada uma UO para cada setor da empresa, reduzindo assim a quantidade de domínios necessários. A Figura 5 exibe uma representação visual de como pode ser feita a organização de objetos em uma UO.

Figura 5 - Exemplo de topologia de Unidades Organizacionais

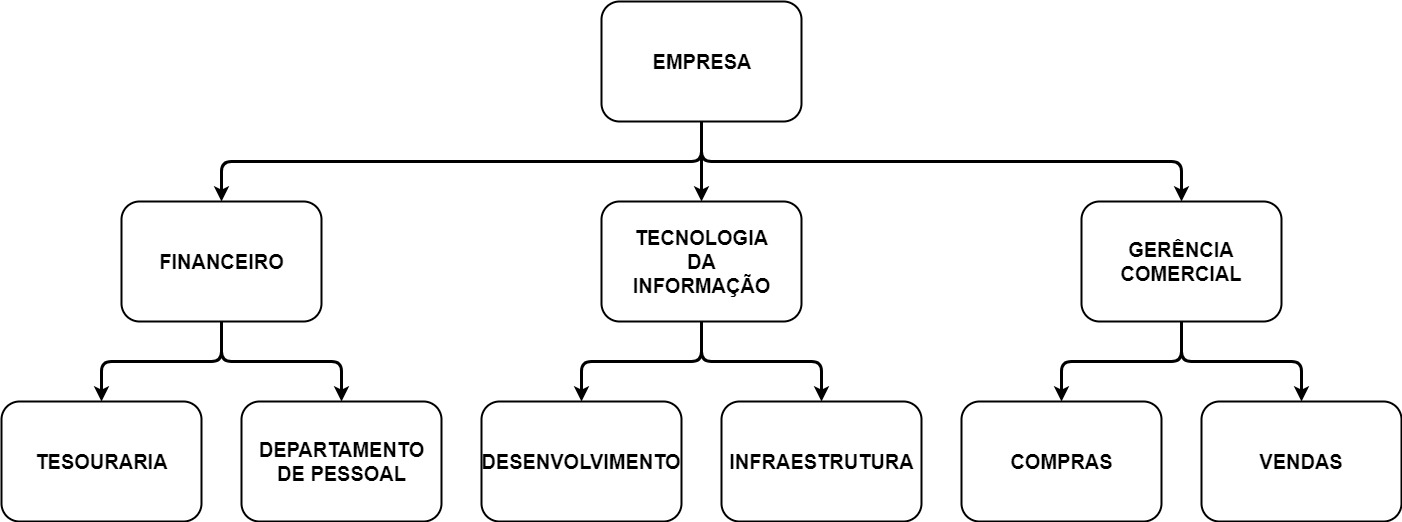
****

Fonte: Autoria própria (2016).

Como mostrado na figura, as UOs ajudam o administrador da rede a organizar melhor os dispositivos que dela fazem parte, na imagem temos a UO\_1 que é exclusiva para impressoras, e a UO\_2 que além conter computadores também inclui outras duas UOs, a UO\_3 que armazena um outro grupo de computadores, e a UO\_4 que é exclusiva para notebooks.

Unidades Organizacionais também podem ser usadas, e são, para definir hierarquia de uma instituição, como empresas, órgãos do governo e etc. Na figura 6, mostrada abaixo, podemos ver um exemplo de como pode ser feito esse tipo de divisão.

Figura 6 - Exemplo de hierarquia utilizando UO



Fonte: Autoria própria (2016).

Outras várias analogias podem ser feitas, como ao esquema de diretórios dos sistemas operacionais, e até mesmo a divisão de um país por estados e cidades.

### 2.3.2 Group Policy Object

GPOs são como modelos (*templates*), existem para agilizar e facilitar o controle de um grande número de objetos em uma rede (tais como perfis de usuários, computadores, etc.). Através dela podemos aplicar uma configuração, restringir uma funcionalidade, automatizar a instalação de softwares, restringir o acesso de usuários a diretórios específicos e outras muitas operações, a uma determinada UO (Unidade Organizacional), ou até mesmo, a um objeto especifico do sistema.

Esses modelos possuem três modos principais de atuação dentro de suas configurações: o *enable*, *disable* e *not configured*.

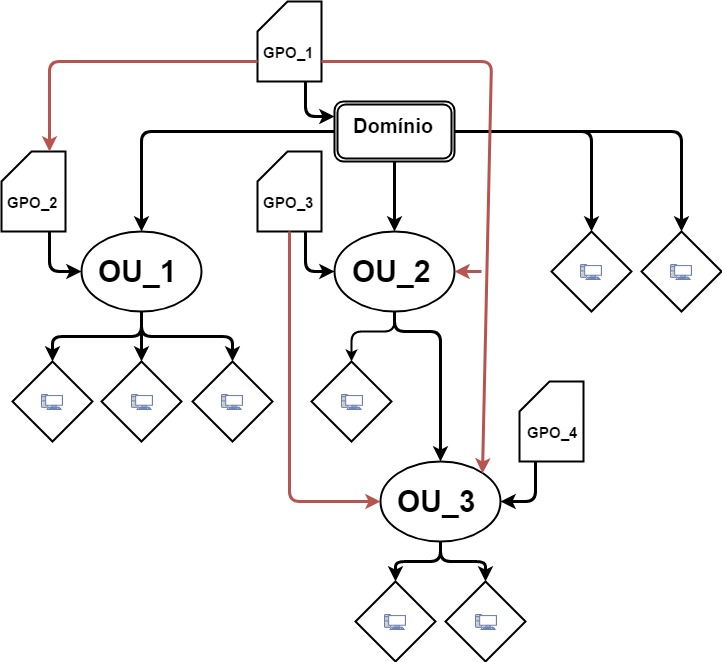
1. *enable*: marcando essa opção a GPO será ativada e passará a exercer a função designada;
2. *disable*: desativará a GPO, ou seja, mesmo se ela estiver atribuída a algum objeto (uma OU, por exemplo), a mesma não surtirá efeito algum;
3. *not configured*: é a configuração padrão, nenhuma alteração na GPO foi feita, ela irá funcionar em configuração *default*.

As GPOs, seguem um modelo hierárquico de três níveis, sendo elas:

1. *sites*: é a representação da estrutura física, ou topologia, de uma rede, sendo o nível mais alto dessa hierarquia. As políticas atribuídas a um *site* serão aplicadas a todos os domínios que dele faça parte, e consequentemente, a tudo que esteja incluído a esses domínios;
2. domínios: O nível intermediário, a política a ele estabelecida, só afetaram o que faz parte desse domínio em especifico, ou seja, domínios que compartilham o mesmo *site* não partilharão das mudanças;
3. unidades organizacionais: são o terceiro e mais baixo nível, um domínio costuma ter várias OUs, mas uma política atrelada a uma OU só afetará ela mesma, deixando de fora as outras OUs do domínio.

As GPOs funcionam com um sistema de herança, aonde um objeto herda políticas atreladas as entidades da qual faz parte indiretamente. Observe a imagem:

Figura 7 - Exemplo de topologia de distribuição de políticas



Fonte: Autoria própria (2016).

A figura 5, ilustra por exemplo, como a OU\_3 além de receber as políticas da GPO\_4, a única a ela atribuída diretamente, ainda herda as políticas da OU\_2 (uma UO pode conter outras UOs dentro) e do Domínio. Ou seja, como a OU\_3 está dentro da OU\_2 e do Domínio, ela vai herdar todas as GPOs atribuídas aos mesmos.

Pode acontecer, por exemplo, de um *site* e uma OU terem duas GPOs distintas com políticas semelhantes, por padrão, o objeto vai herdar a política mais “próxima”, nesse caso, seria a da OU.

## 2.4 HYPERTEXT PREPROCESSOR

O *HyperText Preprocessor* (PHP) foi criado em 1994 pelo programador canadiano-dinamarquês Rasmus Lerdof e é uma linguagem muito utilizada na internet, que permite criar aplicações web mais dinâmicas, possibilitando interações com usuários através do uso de formulários, parâmetros da *Uniform Resource Locator* (URL) e *links*.

Inicialmente, o PHP era usado somente na página pessoal de seu criador, para coletar informações sobre as visitas feitas. Em 1995 é disponibilizada a primeira versão com desenvolvimento colaborativo, e então a linguagem passa a ser conhecida como *Personal Home Page Tools* (nome que deu origem a sigla PHP).

Em meados de 1995 o interpretador foi reescrito, e ganhou o nome de PHP/FI, o “FI” veio de um outro pacote escrito por Rasmus que interpretava dados de formulários HTML (Form Interpreter). Ele combinou os scripts do pacote Personal Home Page Tools com o FI e adicionou suporte a MySQL, nascendo assim o PHP/FI, que cresceu bastante, e as pessoas passaram a contribuir com o projeto (BARRETO, 2000, p.6)

Em 1997 a linguagem deixou de ser somente um projeto pessoal de Lerdof e passou a ser melhorada por um time de desenvolvimento mais organizado, ganhando suporte a orientação de objetos, herança e métodos.

Entre as vantagens do PHP podemos citar a sua simplicidade e facilidade de aprendizado pra novos usuários e a quantidade de recursos para o programador profissional. A sua compatibilidade com diversos bancos de dados de Linguagem de Consulta Estruturada, ou *Structured Query Language* (SQL), tais como MySQL, Oracle e PostgreSQL, e também o suporte a diversos protocolos como *Post Office Protocol* (POP3), *Simple Network Management Protocol* (SNMP) e *Internet Message Access Protocol* (IMAP), tornam o PHP uma ferramenta importante para desenvolvimentos de soluções voltadas a *web*.

## 2.5 ARDUINO

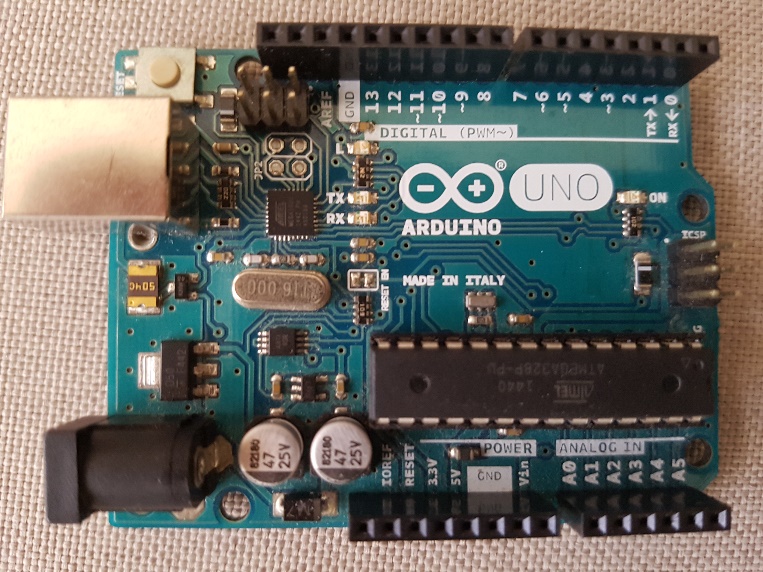
Segundo McRoberts (2011), em termo práticos, um Arduino é um pequeno computador que você pode programar para processar entradas e saídas entre o dispositivo e os componentes externos conectados a ele.

Arduino é uma plataforma *open-source*[[1]](#footnote-1) e pode ser definido como duas coisas, hardware e software. Em questão de hardware, o Arduino é um microcontrolador baseado no chip ATmega328, sendo uma ferramenta usada para diversos fins, como monitoramento de um ambiente (temperatura, umidade, presença) ou automação do mesmo (luz, equipamentos eletrônicos). Com um Arduino, podemos por exemplo, fazer com que as lâmpadas do seu quarto se acendam quando um sensor de presença ligado ao Arduino detecte movimento no cômodo. Ou seja, basicamente, o Arduino é um pequeno computador com a capacidade de conexão com inúmeros sensores (além de botões, displays, *Light Emitting Diode* (LED), pequenos motores e etc) e de receber, processar e enviar os dados recebidos pelos mesmos.

Em termos de software, o Arduino pode ser definido como a *Integrated Development Environment* (IDE) própria para o mesmo, aonde desenvolvemos a aplicação, utilizando uma linguagem de programação bastante similar a linguagem C, mais simples, porém com sintaxe muito similar. O arquivo de código gerado com a IDE Arduino se chama *sketch*, e o *upload* do código para a placa é feita diretamente através dessa interface, bastando apenas o Arduino estar conectado ao computador via cabo USB.

Existem diversos modelos de Arduino, tais como o Uno, Zero, Mega, Nano, Leonardo, Due, M0 Pro, Esplora, Robot, Yún e outros, além também das inúmeras versões clones ou similares/compatíveis, como a *BlackBoard*, Garagino, *Freeduino*, Severino e tantas outras. Qualquer um pode criar sua própria placa clone, pois o código, os esquemas e o projeto estão disponíveis para qualquer um com curiosidade, interesse e conhecimento. A Figura 8 ilustra um dos diversos modelos de Arduino.

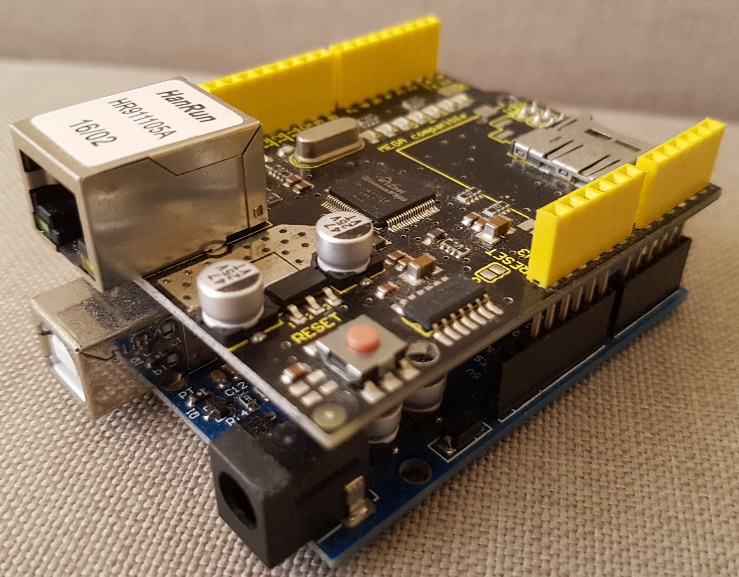
Figura 8 - Arduino Uno, atualmente a versão de Arduino mais popular



Fonte: Autoria própria (2017)

Outro fator positivo da plataforma Arduino, é a vasta quantidade de sensores, *shields* (são como extensões da placa Arduino, agregam alguma funcionalidade e mantém a maioria dos pinos livres), *displays*, LEDs, pequenos motores, botões e etc. que podem ser incorporados a ele. Isso torna a quantidade de projetos possíveis quase que infinitas. Abaixo segue a Figura 9 que demostra como é feito o encaixe da placa Arduino com uma *shield*.

Figura 9 - Arduino Uno integrado a um *shield ethernet*

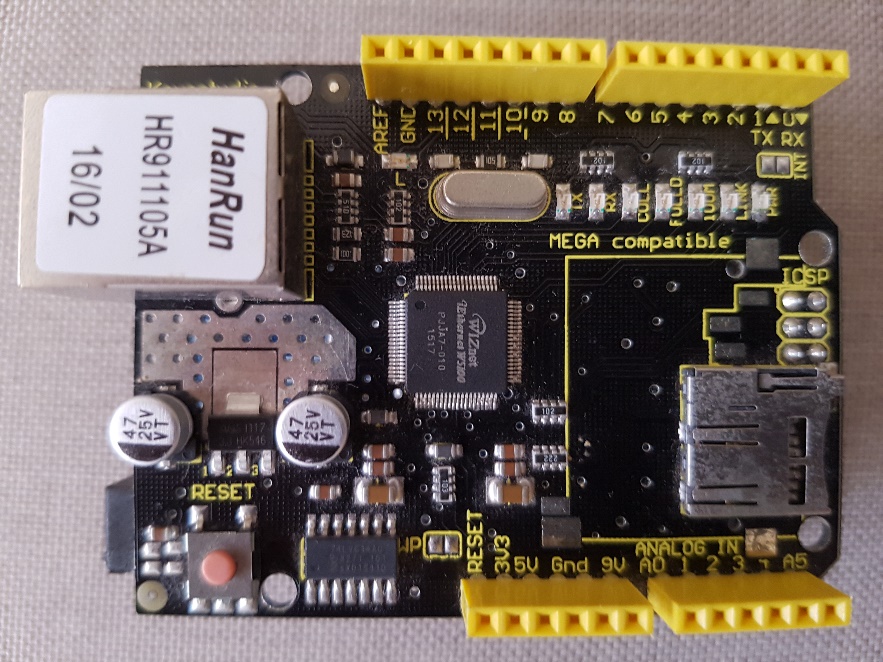


Fonte: Autoria própria (2017).

### 2.5.1 Shield Ethernet W5100

O *Shield Ethernet W5100* é uma placa extensora *open-source* para Arduino, equipada com um *chip* *Wiznet W5100*. Basicamente, provê ao Arduino uma forma de se conectar a *internet*, já que o mesmo não tem a capacidade de se conectar sozinho. O *shield* contém uma porta para cabo RJ-45 e um *slot* para cartão de memória micro SD. A Figura 10 abaixo contém a imagem de uma *shield Ethernet W5100.*

Figura 10 - Shield ethernet equipado com o chip Wiznet W5100



Fonte: Autoria própria (2017).

A conexão acontece via protocolos *Transmission Control Protocol* (TCP) ou *User Datagram Protocol* (UDP) com velocidades de conexão de 10 ou 100mbps e comunicação *full-duplex* ou *half duplex*. A alimentação é de 5V e a placa suporta PoE (*Power over Ethernet*).

A vantagem de um *Shield Ethernet W5100* para um módulo com funcionalidade similar, e que além das vantagens já citadas anteriormente, ele é mais prático de se utilizar e contém um *slot* para cartão micro SD, dando a possibilidade de manter um pequeno servidor com alguns arquivos, como imagens e vídeos.

### 2.5.2 Módulo NFC PN532

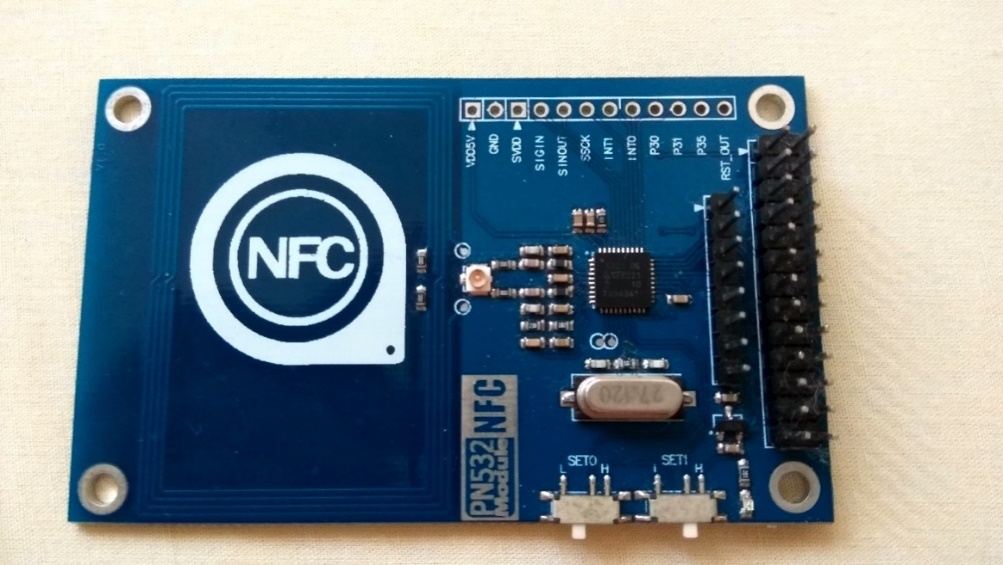
O PN532 é um módulo de transmissão altamente integrado para comunicação sem contato a 13.56MHz, incluindo funcionalidade de microcontrolador baseado em um núcleo 80C51 (NXP SEMICONDUCTORS, 2012).

Seu circuito analógico altamente integrado é capaz de demodular e decodificar respostas, e dependendo do tamanho da antena (existem diversos modelos usando o mesmo chip), o módulo consegue estabelecer comunicação em uma distância de até 5 cm, nas velocidades de 212kbit/s ou 424kbit/.

O PN532 tem suporte aos padrões ISO 14443A e ISO 14443B, sendo compatível com *smart cards* MIFARE e FeliCa.

O módulo precisa de uma fonte de energia de 2.7V a 5.4V para funcionar. Uma imagem do modelo utilizado pode ser vista logo abaixo.

Figura 11 - Modulo PN532



Fonte: Autoria própria (2017).

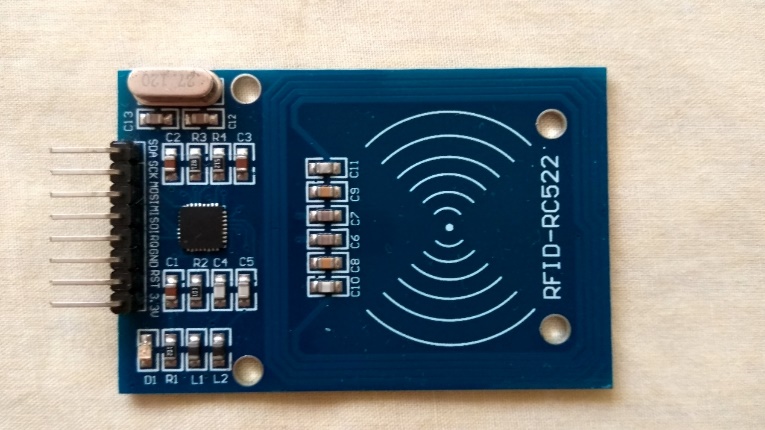
### 2.5.3 Módulo RFID MFRC522

O MFRC522 é um módulo de leitura/escrita altamente integrado para comunicação sem contado a 13,56 MHz (NXP SEMICONDUCTORS, 2016). O módulo suporta *tags* baseadas no padrão ISO/IEC 14443, o mesmo dos modelos MIFARE.

Seu transmissor interno é capaz de realizar operações de leitura e gravação sem necessitar de contato físico com a *tag*.

Entre outras características do MFRC522, podemos citar que ele utiliza o protocolo *Serial Peripheral Interface* (SPI), tem um circuito altamente integrado para demodular e decodificar respostas, suporta uma distância (entre módulo e *tag*) de 5cm, dependendo da antena, para leitura e escrita, e opera com fontes de energia de 2,5V a 3,3V. Uma imagem do modelo utilizado pode ser vista logo abaixo.

Figura 12 - Modulo MFRC522



Fonte: Autoria própria (2017).

# DESENVOVIMENTO DO SISTEMA

Com o intuito de realizar a prova de conceito da solução proposta nesse trabalho, foi desenvolvido um protótipo capaz de identificar *tags* NFC e verificar se as mesmas possuem informações cadastradas em uma base Active Directory, que inclusive já poderia estar operante antes da solução ser implementada. É importante frisar que as tecnologias utilizadas para realizar a integração com o AD são de baixo custo ou *open source*.

## 3.1 PRÉ-REQUISITOS E ARQUITETURA DE DESENVOLVIMENTO

Para a correta integração entre a placa Arduino e os módulos e *shields* é necessário instalar as bibliotecas necessárias. Bibliotecas são trechos de códigos com funções pré-escritas que visão facilitar a vida do desenvolvedor abstraindo parte de um problema. Para o devido funcionamento do projeto em questão, foram utilizadas quatro bibliotecas:

1. pn532.h: Biblioteca usada para executar funções pré-definidas referentes ao módulo PN532;
2. spi.h: Biblioteca necessária para realizar a comunicação entre a placa de prototipagem Arduino e seus módulos;
3. ethernet.h: Biblioteca utilizada para fazer uso das funções referentes a *shield* *Ethernet W5100*;
4. mfrc522.h: Biblioteca necessária para fazer uso das funções pré-definidas para o módulo RFID MFRC522.

A IDE Arduino possui várias bibliotecas instaladas e prontas para o uso, muitas delas para ajudar no aprendizado de quem está utilizando o Arduino pela primeira vez, porém, essas quatro citadas não estão presentes no pacote inicial, todas elas foram baixadas da internet.

O trabalho foi dividido em dois sistemas diferentes, o mais simples que faz somente a escrita nas *tags*, e o mais complexo que realiza tanto a leitura das *tags* como também o envio das informações e o tratamento dos dados recebidos.

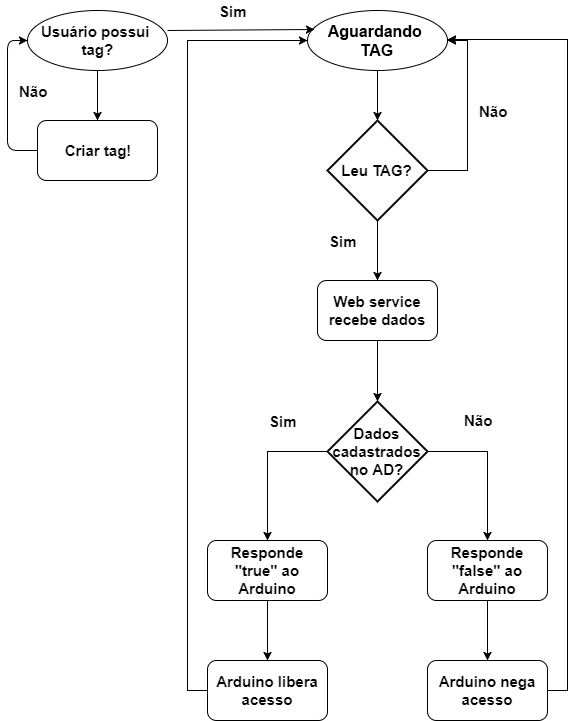
O trabalho teve início com um código baseado no exemplo “readMifareTargetID” da biblioteca “PN532.h”, removendo tudo que era desnecessário, em seguida foi adicionado a *shield* *Ethernet W5100* para enviar e receber dados através da rede. Após isso foi implementado o sistema de diretórios *Active Directory* em uma máquina virtual rodando o *Windows Server 2012*. A próxima etapa foi implementar um *web service* em PHP (rodando em um sistema operacional *Ubuntu Server*) que fosse capaz de receber os dados enviados pelo Arduino e verificar se os mesmos estão presentes no AD, enviando uma resposta ao Arduino de acordo com o resultado obtido na verificação. Sendo comprovado o pleno funcionamento dos códigos implementados no Arduino e no *web service* em PHP, foi iniciado o processo de escrita de informações nas *tags*, para tal foi utilizado o exemplo “rfid\_write\_personal\_data” presente na biblioteca “MFRC522.h”, sendo feita apenas algumas pequenas alterações, no caso, para o código sempre escrever as informações de usuário e senha nos mesmos blocos de memória.

## 3.2 COLETA DE DADOS

O código de autenticação teve um início simples, lendo as *tags* e exibindo suas IDs via monitor serial. Em uma primeira fase, a ID gravada em cada *tag* foi suficiente para dar início a tentativas de autenticação utilizando algum servidor a parte rodando um *web service*. Em seguida foi desenvolvido um *web service* em PHP para fazer a validação das credenciais enviadas pelo Arduino.

Em seguida teve início a busca por uma ferramenta capaz de escrever informações nas *tags*, utilizando os blocos de memória disponíveis para escrita. Foi então utilizado um exemplo disponível na biblioteca do módulo RFID RC522, o “rfid\_write\_personal\_data”.  
 Para finalizar, foi necessário adaptar parte do código presente no exemplo “readAllMemoryBlocks” da biblioteca PN532\_SPI e implementar o mesmo no código de leitura das *tags*. A adaptação se tornou necessária por que o “readAllMemoryBlocks” original lia todos os 64 blocos de memória presente na *tag*, e isso levava muito tempo, tempo esse que tornaria a aplicação inviável. A adaptação foi feita justamente nesse ponto, sendo modificado para ler somente dois blocos específicos, um para armazenar a senha e outra para o nome de usuário, não atrasando assim a execução de todo o código, pois para o usuário a autenticação tem que ser a mais rápida possível. A imagem a seguir contém um organograma que ilustra todas as etapas que o sistema percorre para autenticar uma *tag*.

Figura 13 - Funcionamento do sistema



Fonte: Elaborado pelo autor deste trabalho (2017).

## 3.3 MONTAGEM DO SISTEMA

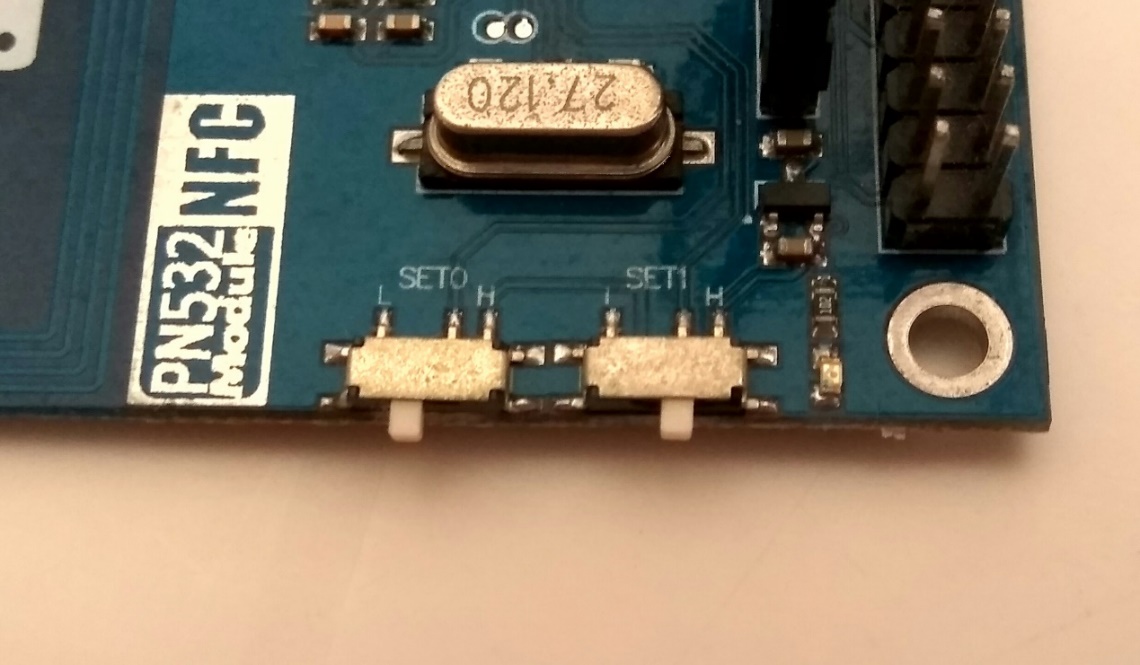
O protótipo de leitura de *tags* foi desenvolvido usando uma *protoboard*, três *Light Emitting Diode* (LED) de cores diferentes (azul, verde e vermelho), três resistores de 100 ohms, treze fios *jumpers*, um módulo NFC PN532 e um Arduino UNO.

O módulo PN532 deve ser conectado ao Arduino seguindo as instruções abaixo:

1. SCK: porta digital 13;
2. MI: portal digital 12;
3. MO/SDA/TX: portal digital 11;
4. NSS/SCL/RX: portal digital 9;
5. IRO: não conectada;
6. RST: não conectada;
7. GND: porta GND do Arduino;
8. 5V: porta 5V do Arduino.

Para funcionar com o Arduino, o módulo PN532 deve estar em modo SPI, para isso é necessário deixar o SET0 em L (*low*) e SET1 em H (*high*) como demostrado na figura a seguir.

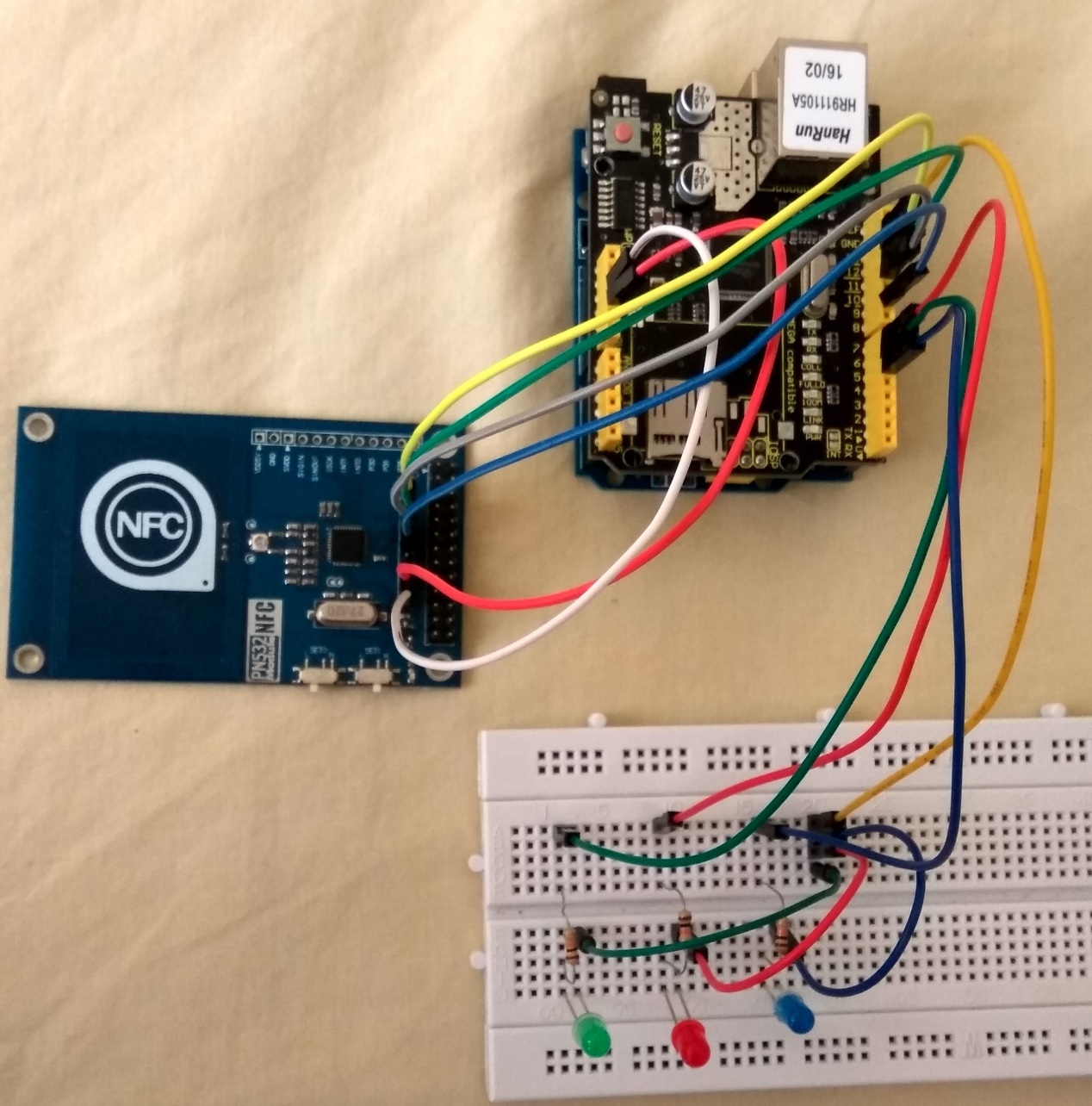
Figura 14 - SET0 e SET1



Fonte: Autoria própria (2017).

Os LEDs foram ligados nas portas digitais 5, 6 e 7 com o auxílio de resistores de 100ohms para evitar que os mesmos queimem. Abaixo se encontra uma foto do protótipo de leitura em sua fase final.

Figura 15 - Protótipo de leitura de tags



Fonte: Autoria própria (2017).

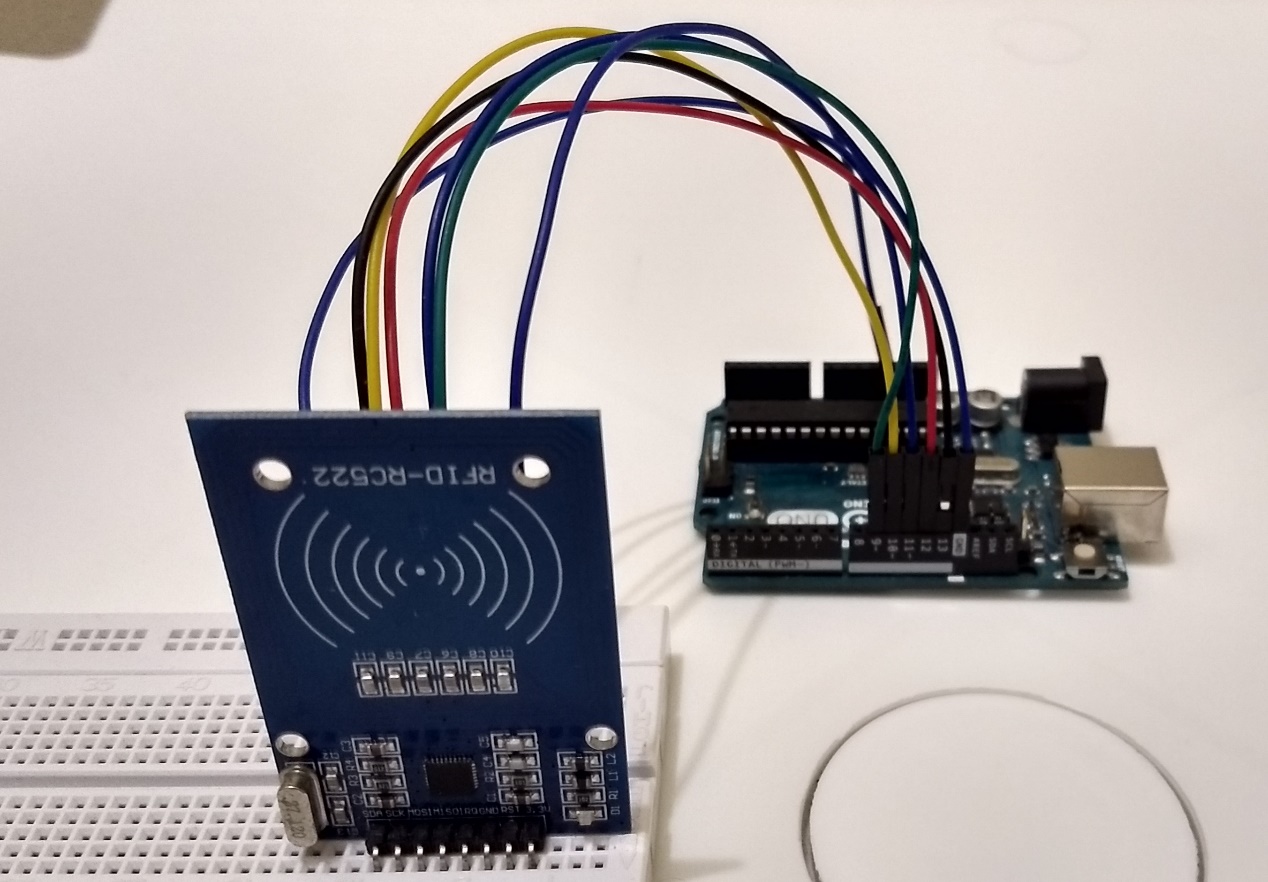
O protótipo usado para gravar informações nas *tags* foi desenvolvido utilizando um módulo RFID MFRC522, uma *protoboard*, sete fios *jumpers* e um Arduino UNO.

Para o funcionamento do módulo MFRC522 em conjunto com o Arduino, é necessário conectar os seguintes pinos:

1. SDA: porta digital 10;
2. SCK: porta digital 13;
3. MOSI: porta digital 11;
4. MISO: porta digital 12;
5. IRQ: não conectado;
6. GND: porta GND do Arduino;
7. RST: porta digital 9;
8. 3.3V: porta 3.3V do Arduino.

Abaixo uma imagem do protótipo utilizado para gravação de informações nas *tags*.

Figura 16 - Protótipo utilizado para gravar usuário e senha nos cartões

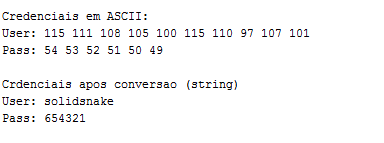


Fonte: Autoria própria (2017).

Abaixo segue algumas particularidades percebidas e aplicadas no desenvolvimento do projeto:

1. o tipo de cartão utilizado foi o MIFARE Classic 1K, com permissões de leitura e escrita;
2. o módulo PN532 conseguiu detectar cartões MIFARE Classic 1K a uma distância aproximada de 4cm. Essa diferença de 1cm entre a distância alcançada nesse projeto e o valor especificado no *datasheet*, pode ser dar devido ao tipo de antena NFC encontrada no módulo utilizado, pois o mesmo aparenta ser um modelo genérico baseado no produzido pela Adafruit;
3. o módulo MFRC522 conseguiu detectar cartões MIFARE Classic 1K a uma distância aproximada de 4cm. Assim como no módulo PN532, essa diferença no alcance real quando comparado as especificações do *datasheet*, pode se dar devido a potência da antena utilizada no produto;
4. por padrão, o pino NSS/SCL/RX do módulo PN532 deve ser conectado a porta digital 10 do Arduino, porém, quando o Arduino se encontra conectado a *shield* *Ethernet W5100* a porta digital 10 passa a ficar bloqueada para uso, pois a mesma é usada para transferência de dados entre o Arduino e a *shield*. Devido a isso, o pino NSS/SCL/RX do módulo PN532 foi conectado a porta digital 9 do Arduino, e adicionado o trecho de código “#define PN532\_CS 9” ao código leitor de *tags*;
5. devido a motivos desconhecidos, as funções da biblioteca “PN532.h” e “Ethernet.h” apresentam algum tipo de conflito, incapacitando o módulo PN532 de continuar a leitura de *tags* após a primeira leitura e tentativa de conexão com o servidor de validação. Para contornar o problema a solução encontrada foi criar uma função do tipo *void* que reinicializa as funções “nfc.begin” e “nfc.SAMConfig”, pertencentes a biblioteca “PN532.h”, sempre que a conexão com o servidor de validação é encerrada;
6. ao ler os blocos de memória das *tags* os valores obtidos estarão em valores do sistema de numeração decimal que representam caracteres da tabela ASCII. Antes de enviar esses dados para o servidor de validação, é necessário realizar uma conversão para o formato *string*. A seguir uma imagem mostrando o antes e o depois de uma consulta as informações armazenadas em uma *tag* MIFARE;

Figura 17 - Dados armazenados em uma tag



Fonte: Autoria própria (2017).

1. o código original de leitura de blocos encontrado no exemplo “readAllMemoryBlocks” faz a leitura de todos os 64 blocos de memória, se mostrando um processo bem lento, levando cerca de 6 segundos para completá-lo. Em uma aplicação real esses 6 segundos adicionais seriam inviáveis, podendo até criar uma fila caso muitos usuários desejassem entrar em uma sala com fechadura eletrônica que fizesse uso desse sistema de autenticação por exemplo. Para contornar tal problema, a solução implementada foi modificar o número de blocos lidos para apenas 2, no caso o bloco 1 e o 4.

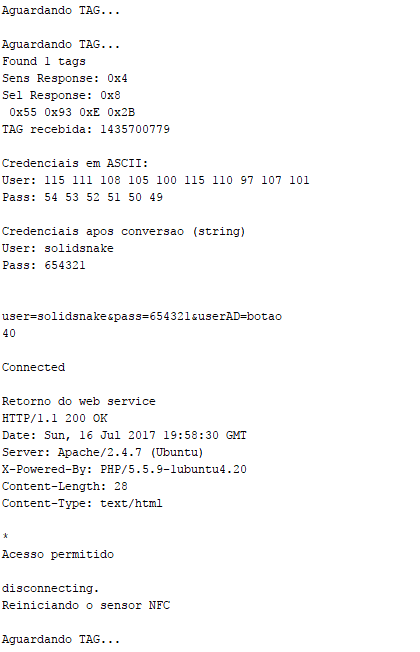
# TESTES E RESULTADOS

Para validarmos o funcionamento do protótipo desenvolvido, foram criados 6 cenários distintos de uso, simulando as situações cotidianas as quais o mesmo seria submetido. O resultado obtido nos testes é comparado com o esperado, juntamente com alguns comentários e o detalhamento sobre cada situação.

## 4.1 CENÁRIO 1

No primeiro cenário foi testado uma *tag* NFC contendo um usuário existente no Active Directory juntamente com a senha definida para tal usuário. O resultado esperado é o acesso aprovado, já que ambas as informações devem estar presentes no AD.  
 O resultado foi conforme o esperado, como podemos ver na Figura 18 (abaixo), o leitor NFC conseguiu capturar o usuário “solidsnake” com a senha “654321” da *tag* NFC, e os mesmos foram validados no AD através do *web service* em PHP.

Figura 18 - Resultado do cenário 1



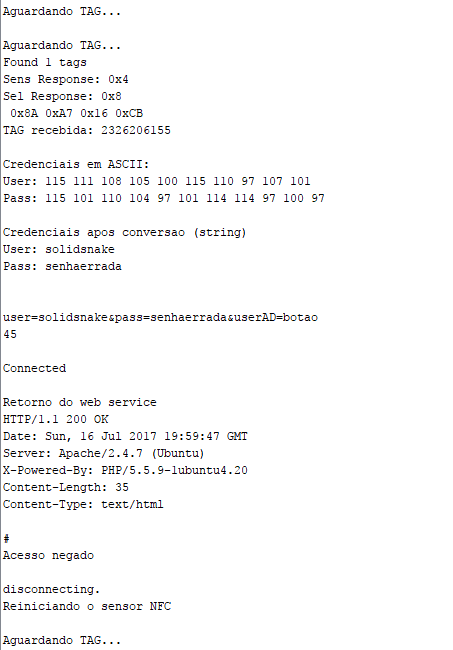
Fonte: Autoria própria (2017).

## 4.2 CENÁRIO 2

No segundo cenário foi utilizado uma *tag* NFC contendo um usuário existente no Active Directory, porém a senha contida no cartão não é igual a armazenada no AD. O resultado esperado é o acesso negado, já que apesar do usuário realmente estar cadastrado no sistema, a informação referente a sua senha, encontrada na *tag*, está errada.

O resultado foi conforme o esperado, a Figura 19 (abaixo), ilustra o que aconteceu, o leitor consegue identificar a *tag*, capturando as informações de usuário e senha nelas contidas e enviando as mesmas para o *web service* fazer a validação, como resposta temos o acesso negado.

Figura 19 - Resultado do cenário 2



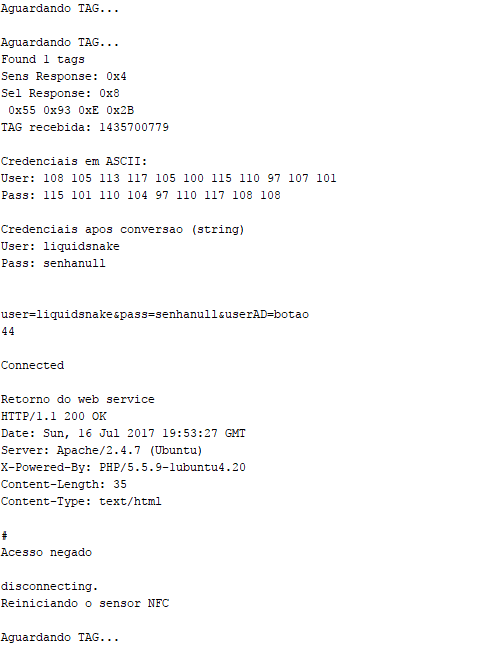
Fonte: Autoria própria (2017).

## 4.3 CENÁRIO 3

No terceiro cenário foi utilizado uma *tag* NFC contendo um usuário que nunca foi cadastrado no AD. O resultado esperado é o acesso negado, pois se o usuário não está cadastrado no AD o mesmo não deve ter permissão para acessar o que está trancando com a fechadura eletrônica.

O resultado apresentado foi o esperado, na Figura 20 (abaixo) podemos ver a tentativa de acesso com uma *tag* contendo um usuário inexistente no AD, como o *web service* não consegue encontrar esse usuário ele acaba barrando o acesso.

Figura 20 - Resultado do cenário 3



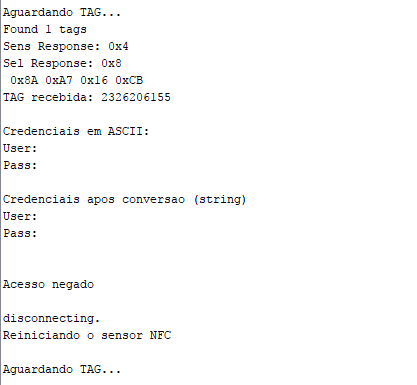
Fonte: Autoria própria (2017).

## 4.4 CENÁRIO 4

No quarto cenário foi utilizado uma *tag* NFC vazia, sem informações de usuário e senha. O resultado esperado é que o sistema barre o acesso devido a *tag* não ter informação alguma para validar.

O resultado alcançado foi conforme o esperado, após verificar que a *tag* está vazia o próprio Arduino nega o acesso, sem necessitar fazer uma requisição ao *web service.* Na figura 21 (abaixo) podemos ver o resultado desse cenário.

Figura 21 - Resultado do cenário 4



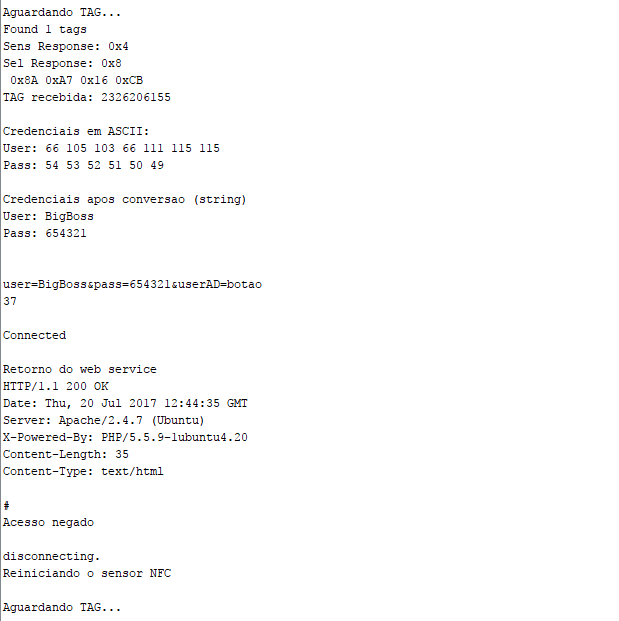
Fonte: Autoria própria (2017).

## 4.5 CENÁRIO 5

No cenário 5 foi testada a possibilidade de um cartão com usuário desabilitado ser usado. O resultado esperado é que o sistema negue o acesso, pois um usuário desabilitado no Active Directory não deve ser válido.

O resultado obtido pode ser conferido na Figura 22 logo abaixo, onde podemos ver que o servidor retorna o acesso negado para o usuário.

Figura 22 - Resultado do cenário 5



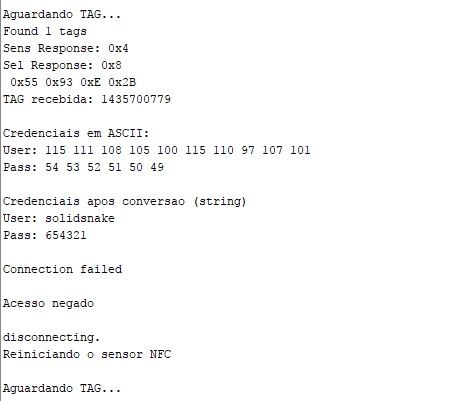
Fonte: Autoria própria (2017).

## 4.6 CENÁRIO 6

O último cenário testado foi o caso do Arduino ou algum dos 2 (dois) servidores ficarem sem conexão de rede, nesse caso o resultado esperado é que o usuário tenha o acesso negado pois o Arduino não vai conseguir validar suas credenciais.

O resultado obtido foi conforme o esperado, como podemos ver na Figura 23 logo abaixo, o Arduino, após capturar as credenciais armazenadas na *tag* NFC, não consegue estabelecer uma conexão com o servidor de validação, negando assim o acesso ao usuário.

Figura 23 - Resultado do cenário 6



Fonte: Autoria própria (2017).

# CONCLUSÃO E TRABALHOS FUTUROS

Este projeto tem como objetivo geral o desenvolvimento de um protótipo tecnológico de baixo custo que substituísse as chaves convencionais. A primeira decisão a ser tomada foi referente a que tipo de chave eletrônica seria usada, e a escolha foi *tags* NFC devido a sua adoção no mercado. Em seguida foi decidido usar um leitor NFC PN532, em conjunto com uma placa de prototipagem Arduino, formando assim o módulo leitor de tags, responsável por obter as informações de credenciais que seriam armazenadas nas *tags*. Para manter a base de dados de credenciais a solução escolhida foi o Active Directory, uma solução proprietária da *Microsoft* baseada em LDAP e largamente utilizada pelas instituições. E no meio foi adotada um *web service* em PHP para realizar a comunicação entre o módulo leitor de *tags* e o AD.

A medida que o desenvolvimento do projeto avançava, foi percebido que somente as ID’s contidas nas *tags* não seriam suficientes para o modelo de autenticação proposto, portanto, foi necessário iniciar a implementação de uma solução capaz de introduzir informações de usuário e senha nas *tags* utilizadas.

O objetivo geral do trabalho conseguiu ser atingido pois, foi possível montar toda a estrutura básica de autenticação, desde a leitura até a verificação das credenciais, além do acréscimo de informações (usuário e senha) nas *tags*, provando ser possível a modernização de uma instituição com um investimento relativamente baixo, já que boa parte do material utilizado no projeto é *open source*.

Como trabalhos futuros seria interessante dar continuidade aos objetivos propostos no início do desenvolvimento do trabalho, mas que não foram concluídos, tais como a implementação do controle de acesso com níveis de permissão para diferenciar usuários. Outro ponto que também não foi finalizado é o registro de acessos, atualmente nenhuma informação, referente as tentativas de autenticação, é armazenada.

Uma implementação importante e que agregaria muito ao trabalho, seria um *software* de gerenciamento da solução, algumas funcionalidades interessantes para essa possível ferramenta seriam: possibilidade de gravar informações nas *tags* (dispensando o uso do monitor serial do Arduino para isso) e uma listagem dos acessos realizados, podendo ser filtrado por usuário, data e hora e local.

E por último, até a conclusão deste trabalho, o mesmo não conta com uma solução eficiente para questões de segurança, as informações nas *tags* são facilmente acessadas, inclusive por *smartphones*, e é igualmente fácil modificá-las ou cloná-las. Uma solução a curto prazo seria associar as informações inseridas nas *tags* com as IDs das mesmas, evitando que estas sejam facilmente clonadas.

# REFERÊNCIAS

APPLE Pay. In: **APPLE. Califórnia.**Disponível em: <http://www.apple.com/apple-pay/>. Acesso em: 16 mar. 2016.

**IOS Feature Availability.** In: APPLE. **Califórnia.**Disponível em: <https://www.apple.com/ios/feature-availability/#apple-pay>. Acesso em: 09 abr. 2017.

BHAT, Pali. **Android Pay now in the UK, new countries on the way. 2016.**Disponível em: <https://www.blog.google/products/android/android-pay-now-in-uk-new-countries-on/>. Acesso em: 08 abr. 2017.

CLARK, Sarah. **VingCard launches NFC room key system for hotels:**The hotel room key market leader, part of the Assa Abloy group which recently ran an NFC trial at the Clarion Hotel Stockholm, has now brought the technology to market. 2011. Disponível em: <https://www.nfcworld.com/2011/06/28/38366/vingcard-launches-nfc-room-key-system-for-hotels/>. Acesso em: 10 maio 2017.

COSKUN, Vedat; OK, Kerem; OZDENIZCI, Busra. **Near Field Communication:**From Theory To Practice. Chichester: Wiley, 2012. 390 p.

FERRAZ FILHO, Onildo Luciano de Souza. **Comunicação NFC (Near Field Communication) entre Dispositivos Ativos.**2010. 100 f. Monografia (Especialização em Engenharia da Computação) - Universidade Federal de Pernambuco, Recife, 2010.

G1. **Samsung Pay, meio de pagamento por celular, chega ao Brasil:**Cartões de BB, Caixa e Santander já rodam o sistema de pagamento digital. É possível cadastrar cartões de crédito e débito para fazer transações, 2016. Disponível em: <http://g1.globo.com/tecnologia/noticia/2016/07/samsung-pay-meio-de-pagamento-por-celular-chega-ao-brasil.html>. Acesso em: 08 abr. 2017.

**ANDROID Pay:** Pay your way. In: GOOGLE. Califórnia. Disponível em: <https://www.android.com/pay/>. Acesso em: 16 mar. 2016.

JURGENSEN, Timothy M.; GUTHERY, Scott B.. **Smart Cards:**The Developer's Toolkit. Nova Jérsei: Prentice Hall, 2002. 432 p.

MCROBERTS, Michael. **Arduino Básico.**2. ed. São Paulo: Novatec, 2011. 456 p.

10 Motivos para Escolher o MySQL para Aplicativos Web, 2016. MYSQL. Califórnia. Disponível em: <https://www.mysql.com/why-mysql/white-papers/top-10-reasons-to-choose-mysql-for-web-based-applications-pt/>. Acesso em: 11 mar. 2016.

MYSQL in Government, 2016. MYSQL. Califórnia. Disponível em: <https://www.mysql.com/industry/government/>. Acesso em: 11 mar. 2016.

Newshawk Time. Smart Card Market Report Aims To Outline and Forecast, Organization Sizes, Top Vendors and End Users Analysis By 2021. Disponível em: <http://newshawktime.com/smart-card-market-report-aims-to-outline-and-forecast-organization-sizes-top-vendors-and-end-user-analysis-by-2021/>. Acesso em: 03 dez. 2017.

**ABOUT the Technology.**NFC FORUM. Disponível em: <http://nfc-forum.org/what-is-nfc/about-the-technology/>. Acesso em: 02 mar. 2016.

NFC World. **Carta offers turnkey kit for NFC payments pilots**. Disponível em: <https://www.nfcworld.com/2013/04/17/323578/carta-offers-turnkey-kit-for-nfc-payments-pilots/>. Acesso em: 03 dez. 2017.

WAYS to Use Near Field Communication. NFC.ORG. Disponível em: <http://nearfieldcommunication.org/using-nfc.html>. Acesso em: 08 abr. 2017.

NXP SEMICONDUCTORS. **MIFARE Classic EV1 1K:**Mainstream Contactless Smart Card IC for Fast and Easy Solution Development. Eindhoven. 2014. 40 p. Disponível em: <http://www.nxp.com/docs/en/data-sheet/MF1S50YYX\_V1.pdf>. Acesso em: 15 jun. 2017.

NXP SEMICONDUCTORS. **PN532/C1:** Near Field Communication (NFC) controller. Eindhoven. 2012. 27 p. Disponível em: <http://www.nxp.com/docs/en/data-sheet/PN532\_C1\_SDS.pdf>. Acesso em: 10 jan. 2017.

NXP SEMICONDUCTORS. **MFRC522:** Standard performance MIFARE and NTAG frontend. Eindhoven. 2016. 95 p. Disponível em: < https://www.nxp.com/docs/en/data-sheet/MFRC522.pdf>. Acesso em: 10 jan. 2017.

**MIFARE ICs:** Overview. In: NXP SEMICONDUCTORS. Eindhoven. Disponível em: <http://www.nxp.com/products/identification-and-security/smart-card-ics/mifare-ics:MC\_53422>. Acesso em: 05 mar. 2016.

**NFC Everywhere.**In: NXP SEMICONDUCTORS. Eindhoven. Disponível em: <http://www.nxp.com/products/identification-and-security/nfc-and-reader-ics/nfc-everywhere:NFC-TECHNOLOGY>. Acesso em: 01 mar. 2016.

NXP SEMICONDUCTORS. **Tap into MIFARE:**The leading contactless platform for launching your next great idea. Eindhoven. 2015. 12 p. Disponível em: <http://www.nxp.com/documents/brochure/About\_MIFARE\_Brochure\_final.pdf>. Acesso em: 05 mar. 2016.

**3 reasons hotels are switching to NFC,**2015. NXP SEMICONDUCTORS. Eindhoven. Disponível em: <https://blog.nxp.com/identification-and-security/three-reasons-hotels-are-switching-to-nfc>. Acesso em: 10 maio 2017.

RFID Insider. **Everything You Need to Know About NFC Smart Posters**. Disponível em: <https://blog.atlasrfidstore.com/nfc-smart-posters>. Acesso em: 03 dez. 2017.

SMARTCARD e Javacard: Arquitetura, 2015. Disponível em: <https://www.gta.ufrj.br/ensino/eel879/trabalhos\_vf\_2015\_2/Seguranca/conteudo/Smartcard-e-Javacard/Arquitetura.html>. Acesso em: 07 jul. 2017.

**FELICA:**Contactless IC Card Technology. SONY Global. Tóquio. Disponível em: <http://www.sony.net/Products/felica/>. Acesso em: 16 mar. 2016.

**WHAT is FeliCa?:**"Contactless" convenience with Sony's FeliCa. SONY Global. Tóquio. Disponível em: <http://www.sony.net/Products/felica/about/>. Acesso em: 05 mar. 2016.

STANEK, William R.. **Windows Server 2008:**Guia Completo. Porto Alegre: Bookman, 2009. 1519 p.

STMICROELECTRONICS. **TN1216 Technical Note:**ST25 NFC Guide. 2. ed. Genebra: Stmicroelectronics, 2016. 38 p. Disponível em: <http://www.st.com/content/ccc/resource/technical/document/technical\_note/f9/a8/5a/0f/61/bf/42/29/DM00190233.pdf/files/DM00190233.pdf/jcr:content/translations/en.DM00190233.pdf>. Acesso em: 15 maio 2017.

TECHTUDO. **Como usar o NFC do seu smartphone?**. Disponível em: <http://www.techtudo.com.br/dicas-e-tutoriais/noticia/2013/12/como-usar-o-nfc-do-seu-smartphone.html>. Acesso em: 03 dez. 2017.

THOMPSON, Marco Aurélio da Silva. **Windows Server 2012:**Instalação, Configuração e Administração de Redes. 2. ed. São Paulo: Érica, 2013. 368 p.

UNICAMP (São Paulo). **Cartão de Identidade Institucional:**Informações Técnicas. Disponível em: <https://www.smartcard.unicamp.br/?page\_id=254>. Acesso em: 7 jul. 2017.

WALSHAW, Robin. **Mission Critical Windows 2000:**Server Administration. Rockland: Syngress, 2000. 560 p.

# APÊNDICE A – CÓDIGO FONTE DO SISTEMA

//Sistema de leitura de TAGs

//Autor: Newton Teixeira de Brito Neto

//Versão: 6.4

//Ano: 2017

#include <PN532.h> //Biblioteca do módulo PN532

#include <SPI.h> //Biblioteca de Interface Periférica Serial

#include <Ethernet.h> //Biblioteca da shield Ethernet W5100

//Configurações do leitor NFC

#define PN532\_CS 9

PN532 nfc(PN532\_CS);

//Configuração de rede

byte mac[] = { 0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xED };

IPAddress ip\_shield(192, 168, 0, 105);

IPAddress ip\_server(192, 168, 0, 102);

EthernetClient client;

//Variaveis auxiliares

String dados\_requisicao;

int tamanho\_requisicao;

int armazenaConnect = 0;

int armazenaAvailableBytes = 0;

char armazenaUser[16];

char armazenaPass[16];

int contaTrue = 0;

int blockn;

String stringUser;

String stringPass;

//Inicializa o leitor NFC

void inicializar(){

nfc.begin();

nfc.SAMConfig();

}

void setup(void) {

Serial.begin(9600);

Ethernet.begin(mac, ip\_shield); //Iniciando as configurações de IP da shield

inicializar(); //Chama a função que inicializa o leitor NFC

//Define em quais pinos os LEDs estão conectados

pinMode(5, OUTPUT);

pinMode(6, OUTPUT);

pinMode(7, OUTPUT);

}

void loop(void) {

//Inicializa os LEDs, mas de inicio apenas o do pino 5 irá acender

digitalWrite(5, HIGH);

digitalWrite(6, LOW);

digitalWrite(7, LOW);

//Faz uma leitura inicial da TAG e exibe informações sobre a mesma, ID em hexadeciamal

uint32\_t id;

id = nfc.readPassiveTargetID(PN532\_MIFARE\_ISO14443A);

//Enquanto o leitor não identificar uma TAG o código vai retornar pro inicio do loop

if (id == 0) {

Serial.println();

Serial.println("Aguardando TAG...");

return;

}

//Ao ler uma TAG o código vai entrar nesse IF

if (id != 0) {

Serial.println();

Serial.print("TAG recebida: ");

Serial.println(id);

Serial.println();

contaTrue = 0;

digitalWrite(5, LOW);

//Leitura dos blocos de memoria das TAGs (buscando informações de usuário e senha)

uint8\_t keys[]= { 0xFF,0xFF,0xFF,0xFF,0xFF,0xFF };

blockn = 1;

if(nfc.authenticateBlock(1, id ,blockn,KEY\_A,keys)) {

uint8\_t block[16];

if(nfc.readMemoryBlock(1,blockn,block)) {

for(uint8\_t i=0;i<16;i++) {

if (block[i] == 32) {

break;

}

char c = (char) block[i];

armazenaUser[i] = c;

}

}

}

blockn = 4;

if(nfc.authenticateBlock(1, id ,blockn,KEY\_A,keys)) {

uint8\_t block[16];

if(nfc.readMemoryBlock(1,blockn,block)) {

for(uint8\_t i=0;i<16;i++) {

if (block[i] == 32) {

break;

}

char c = (char) block[i];

armazenaPass[i] = c;

}

}

}

Serial.println("");

stringUser = String(armazenaUser);

stringPass = String(armazenaPass);

Serial.println(stringUser);

Serial.println(stringPass);

memset(armazenaUser, 0, sizeof(armazenaUser));

memset(armazenaPass, 0, sizeof(armazenaPass));

//A execução terá continuidade caso as informações necessárias tenham sido encontradas nos blocos 1 e 4

if (stringUser != 0 && stringPass != 0) {

//Conectando ao servidor e armazenando o resultado da conexão (1, -1, -2, -3, -4), o ideal é 1

armazenaConnect = client.connect(ip\_server, 80);

//Caso a conexão seja bem sucedida os dados vão ser enviados via POST para o servidor

if (armazenaConnect == 1) {

//Montando a solicitação POST

dados\_requisicao="";

dados\_requisicao.concat("user=");

dados\_requisicao.concat(stringUser);

dados\_requisicao.concat("&pass=");

dados\_requisicao.concat(stringPass);

dados\_requisicao.concat("&userAD=botao");

stringUser = "";

stringPass = "";

tamanho\_requisicao = dados\_requisicao.length();

Serial.println();

Serial.println(dados\_requisicao);

Serial.println(tamanho\_requisicao);

Serial.println();

Serial.println("Connected");

client.println("POST /Z-AD/AD.php HTTP/1.1");

client.print("Host: ");

client.println(ip\_server);

//client.println("Connection: keep-alive");

client.println("Content-Type: application/x-www-form-urlencoded");

client.print("Content-Length: ");

client.println(tamanho\_requisicao);

client.println();

client.println(dados\_requisicao);

client.println("Connection: close");

client.println();

armazenaConnect = 0;

//Recebendo o retorno do servidor

Serial.println();

Serial.print("Retorno do web service");

Serial.println();

//Armazena a quantidade de bytes recebidos

armazenaAvailableBytes = client.available();

//Logica para exibir o conteudo recebido

while (armazenaAvailableBytes > 0) {

char c = client.read();

if (contaTrue < 4) {

Serial.print(c);

}

if (c == '\*') {

contaTrue = 4;

}

if (c == '#') {

contaTrue = 5;

}

armazenaAvailableBytes = armazenaAvailableBytes - 1;

}

}

else {

Serial.println("Connection failed");

}

}

}

if (contaTrue == 4){

Serial.println();

Serial.println("Acesso permitido");

//Serial.println(y);

Serial.println();

contaTrue = 0;

digitalWrite(6, HIGH);

}

else {

Serial.println();

Serial.println("Acesso negado");

Serial.println();

contaTrue = 0;

digitalWrite(7, HIGH);

}

Serial.println("disconnecting.");

client.stop(); //Encerra a conexão

Serial.println("Reiniciando o sensor NFC");

inicializar(); //Reinicializa a placa NFC

delay(3000);

}

# APÊNDICE B – CÓDIGO FONTE DO SISTEMA

//Web service para validar credenciais

//Autor: Newton Teixeira de Brito Neto

//Versão: 3.0

//Ano: 2017

<?PHP

if ($\_POST['aplicar'] == "Botao 1"){

echo $\_POST['name'];

echo '<br>';

echo $\_POST['passwd'];

}

//Função para validar as credenciais no AD

if ($\_POST['userAD']){

function valida\_ldap($srv, $usr, $pwd) {

$ldap\_server = $srv;

$auth\_user = $usr;

$auth\_pass = $pwd;

//Tenta se conectar com o servidor

if (!($connect = @ldap\_connect($ldap\_server))) {

return FALSE;

}

//Tenta autenticar no servidor

if (!($bind = @ldap\_bind($connect, $auth\_user, $auth\_pass))) {

// se não validar retorna false

return FALSE;

}

else {

//se validar retorna true

return TRUE;

}

}

$server = "192.168.0.12"; //IP ou nome do servidor

$dominio = "@dominet.ifrn.br";

$user = $\_POST['user'].$dominio;

$pass = $\_POST['pass'];

//Se o campo referente a senha estiver vazio ele será preenchido com um valor qualquer

if ($pass==""){

$pass=1000;

}

if (valida\_ldap($server, $user, $pass)) {

echo "\* true - usuario autenticado";

}

else {

echo "# false - usuario ou senha invalida";

}

}

?>

1. do inglês, significa código aberto, é um termo usado para definir software ou hardware que tem seu código fonte e esquema divulgado, permitindo que outros usuários o modifiquem [↑](#footnote-ref-1)