

**MARCOS ROCHA PEREIRA**

**PEDRO LUIZ CASSIANO DE CAMPOS**

CONTROLE DE ACESSO: UTILIZANDO RASPBERRY PI E TECNOLOGIAS DE BAIXO CUSTO

SÃO CAETANO DO SUL/ SP

2019



**MARCOS ROCHA PEREIRA**

**PEDRO LUIZ CASSIANO DE CAMPOS**

CONTROLE DE ACESSO: UTILIZANDO RASPBERRY PI E TECNOLOGIAS DE BAIXO CUSTO

Trabalho de Conclusão de Curso apresentado à Faculdade de Tecnologia de São Caetano do Sul, sob a orientação do Prof. Ismael Parede, como requisito parcial para a obtenção do diploma de Graduação no Curso Tecnológico em Segurança da Informação.

SÃO CAETANO DO SUL/ SP

2019

RESUMO

Este trabalho tem como objetivo, demonstrar um projeto de sistema embarcado na placa Raspberry PI e integra-la a um microcontrolador Arduíno. Este projeto descreve a experiência obtida com desenvolvimento de aplicações para equipamentos de segurança. No projeto foram aplicados estudos práticos, normas de segurança como a ISO 27001 e 27002. O sistema foi desenvolvido em várias linguagens como Shell Script, Python, banco de dados Mysql em sua integração do uma interface web Bootstrap. Queremos como objetivo aprimorar algumas ideias já existentes e criar um equipamento de controle de acesso com tecnologias de baixo custo que serão aplicadas ao mercado de segurança patrimonial. Por fim o projeto apresentará testes de segurança, estudos sobre hardening, boas práticas de segurança física e a boas práticas na hora de realização de Pentestes no equipamento.

Palavras-chaves: Controle de Acesso, Raspberry PI, Sistemas Embarcado.

ABSTRACT

This paper aims to demonstrate a system design embedded in the Raspberry PI board and integrated with Arduino. This project will describe our experience with developing applications for security equipment. In the project were applied practical studies and the application of security standards such as ISO 27001 and 27002. The system was developed in several languages such as Shell Script, Python, System integration with a web interface and a Mysql database. We aim to improve some ideas and create access control equipment with low cost technologies that will be applied to the security market. Finally, the project will present safety tests, studies on safety hardening and good physical safety practices and good practices when performing a pentest on this type of equipment.

Keywords: Access Control, Raspberry PI, Embedded Systems.

**Sumário**

1. O Sistema de Controle de Acesso......................................................................11
   1. O sistema lógico do controle de acesso................................................14
2. . O que é a placa Raspberry PI e quais módulos serão utilizados no projeto....17
3. Conclusão do Projeto..........................................................................................53

Considerações Finais..........................................................................................55

Referências.........................................................................................................60

**Introdução**

1. **Apresentação do Tema**

O controle de acesso em determinados locais hoje em dia é algo de fundamental e vital importância. Pensando nesse tema, será elaborado um projeto que visa criar um sistema eficiente no controle de acesso e com tecnologia de baixo custo no qual é possível agilizar e liberar o acesso de pessoas autorizadas.

Segundo a norma ABNT NBR 27002 (2013, p. 04), “A segurança da informação é alcançada pela implementação de um conjunto adequado de controles, incluindo políticas, processos, procedimentos, estrutura organizacional e funções de software e hardware”. Partindo dessa informação da ISO 27002, iremos adaptar algumas normas de segurança no sistema e no dispositivo.

O projeto consiste na criação de um equipamento para o controle de acesso em um ambiente. Implementaremos um sistema de autenticação, com o diferencial de que o produto terá uma rápida resposta no processo de validação com um duplo fator no qual é utilizado a cartão NFC e uma senha personalizada que ficaram disponíveis em uma base de dados local.

A principal ideia é seguir padrões e normas de segurança para criação e desenvolvimento do projeto. Nesse projeto será demonstrado de forma técnica e prática com o uso de parte do conhecimento de segurança da informação que foi aprendido durante curso de graduação.

Escolhemos como base do projeto o Raspberry PI que é um microcomputador de custo acessível, fácil de ser implementado, flexível e podendo ser projetado desde um nível simples para quem tem pouca ou nenhuma experiência, até mesmo com um grau maior de complexibilidade. O intuito desenvolver um controle de acesso embarcado no Raspberry PI.

O dispositivo consistirá num sistema automatizado de autenticação e com validação de acesso, para que somente pessoas autorizadas acessem um determinado ambiente físico.

A escolha do Raspberry PI consiste em entregar um projeto de baixo custo que será possível adicionar este sistema em uma rede para empresas de pequeno porte, para que atenda suas necessidades básicas no controle de acesso de um ambiente.

Sobre o Raspberry PI, Richardson (2013) exalta a simplicidade da Raspeberry PI, pois ele possui as qualidades de ser pequeno barato, hackeável e educacional. Ele possui essas características, pois foi produzido para ser acessível para qualquer um.

O Raspberry PI tem um suporte ideal para a modulagem, ele pode ser integrado com módulos (peças que integram a placa física), instalado em um sistema de automação e pode ser moldado conforme a necessidade do cliente.

1. **Objetivo**

O objetivo é a criação de um equipamento que controle o acesso de pessoas nos mais diferentes ambientes, utilizaremos tecnologias de baixo custo para que seja acessível desde uma empresa de pequeno a médio porte. O dispositivo que estamos projetando, poderá ser instalado em locais de baixa ou alta movimentação onde exista a necessidade de um controle no acesso, o diferencial é que não necessite da intervenção humana, para tal procedimento.

Como objetivo principal demonstraremos que com o uso de uma tecnologia acessível é possível automatizar um processo tornando-o seguro, gerar maior agilidade e proporcionar segurança no ambiente corporativo. O dispositivo funcionará com uma catraca que permitirá o acesso de pessoas que tiverem a sua credencial cadastrada num banco de dados local ou remoto, podemos usar como exemplo um cartão de acesso NFC, autenticação NFC do próprio celular que possui identificação única e complementando existirá uma senha, tudo isso acompanhado de monitoramento que será implementado na sala de segurança, no qual é exibida a imagem do proprietário da identificação. Assim adicionando uma camada de segurança na aplicação que apresentará maior confiabilidade em quem está no local.

O objetivo específico é criar um equipamento com o sistema de controle de acesso embarcado, isso proporcionará uma maior eficiência e maior agilidade no processo de autorização e acesso a pessoas; tornando-o mais seguro o ambiente, utilizando boas práticas descritos nas normas de segurança.

1. **Problematização**

As questões do projeto:

1. A pergunta que é feita é possível implementar um serviço e um bom de controle de acesso de forma segura, baixo custo de implementação e utilizando tais tecnologias?
2. Quais as melhores práticas de segurança que serão aplicadas no projeto?
3. Por que o Raspberry PI?
4. Quais Normas de Segurança da Informação mais se adequaram ao projeto?
5. **Justificativa**

O projeto possibilita o desafio de implementar uma solução segura de controle de acesso com Raspberry PI. Segundo Eben Upton (2014, p. 26) “A Raspberry Pi é uma máquina completa e com considerável poder de processamento e uma placa de circuito impresso menor do que um cartão de crédito”. O grande desafio é poder criar um equipamento diferenciado, com baixo custo de implantação e possibilitando a todos os tipos de empresa a utilização deste equipamento.

O mercado de equipamentos nesta área não chama muito a atenção das empresas de pequeno e médio porte, por causa do alto custo e das soluções que algumas empresas oferecem. Empresas como a Intelbras, a DIMEP e MADIS que são reconhecidas no mercado, oferecem soluções de alto valor de mercado e que acabam desencorajando empresas de pequeno porte.

Pesquisamos em site de empresas do ramo alguns equipamentos e sistemas chegam a custar de R$ 1.000,00 a R$ 5.000,00. Verificamos que alguns equipamentos de baixo custo até atendem de forma simples o controle de acesso que algumas empresas necessitam. O projeto é uma possível solução com tecnologias atuais e com maior margem de segurança.

A ideia é criar um equipamento de fácil instalação, que pode ser gerenciado por pessoas que possuam pouco conhecimento na área, criaremos um equipamento de manuseio simples, sua forma de implementar segurança que utilizará padrões baseados em tecnologia e comunicação.

Temos o intuito de prover um modelo exclusivo e voltado todos os tipos de clientes. O que queremos é oferecer um equipamento que permita melhorar um processo já existente, que pode ser aperfeiçoado com a automatização um ambiente, ainda por cima, facilitar a comunicação entre pessoas e tecnologias com foco em segurança.

1. **Metodologia**

A metodologia de forma qualitativa e de forma exploratória através um ambiente de testes físico e virtual no qual será exportado para um ambiente de produção para realizarmos testes e sua validação. Para o ambiente de testes foi utilizado um software desenvolvido por nós utilizando linguagem PYTHON, BASH e interface gráfica BOOTSTRAP. Para a base metodológica do trabalho usamos as seguintes fontes:

* Leitura de artigos da área de tecnologia da informação para a elaboração do sistema.
* Leitura e citações de livros e projetos especializados na utilização de microcomputadores como o Raspberry PI.

1. **Resultados Esperados**

Como resultados, esperamos obter um projeto confiável e que conheceremos profundamente para que seja possível ajudar a solucionar quaisquer erros ou falhas humanas e que venham a ser encontradas. E esperamos buscar mais conhecimento no desenvolvimento do equipamento. Por fim o projeto pode se transformar em um produto de aceitação no mercado ou no meio acadêmico para estudos com o Raspberry PI.

* + - 1. **O sistema de Controle de acesso com Raspberry PI.**

Primeira uma definição simples do que é controle de acesso físico, segundo Coelho (2014, p. 127):

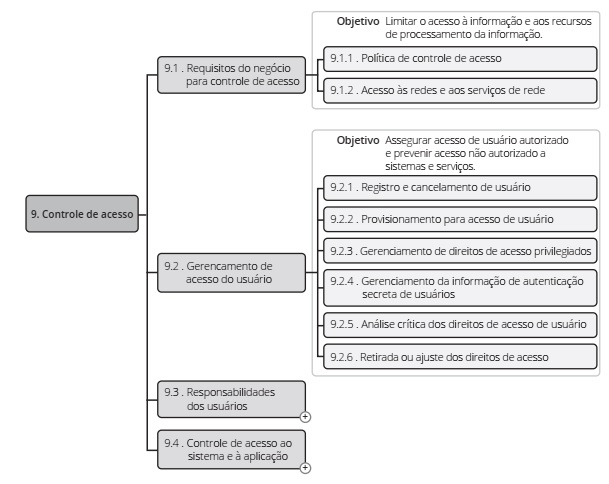
Os controles de acesso físico têm o objetivo de aplicar medidas preventivas para proteger equipamentos, documentações, suprimentos e informações contra acessos não autorizados, perdas etc. Sendo assim, tais controles servem como uma barreira adicional de segurança para o controle de acesso lógico.

Sistema de controle de acesso lógico é um sistema que controla o acesso, a um determinado local e permite que um indivíduo tenha acesso a determinado sistema podendo ser via tecnologia ou acesso a um ambiente físico.

Um sistema de controle de acesso poderá realizar o registro e o gerenciamento detalhado de todo o fluxo de pessoas, atualmente dispositivos de identificação, sistemas sendo por senhas, leitores de chaveiro (tags NFC), e outros por controle remoto ou impressão digital. O sistema registra toda passagem por suas barreiras (catracas) e assim fica registrado tudo em um banco de dados na forma de logs.

Nos dias atuais foram criados vários meios de comunicação e tecnologias para garantir a segurança de um ambiente e toda essa tecnologia pode ser implementada e automatizada. O projeto que será produzido será adaptado segundo a ISO 27001 e ISO 27002. A ideia é fazermos a aplicação seguindo as melhores práticas de gerenciamento de segurança, melhores práticas de para implementação de segurança física e do ambiente e melhores práticas para se ter um bom controle de acesso. A figura 1 exemplifica detalhadamente como iremos utilizar no projeto

Figura 1. Seção 9 da ISO 27002:2013



Fonte: Flavia Coelho (2014)

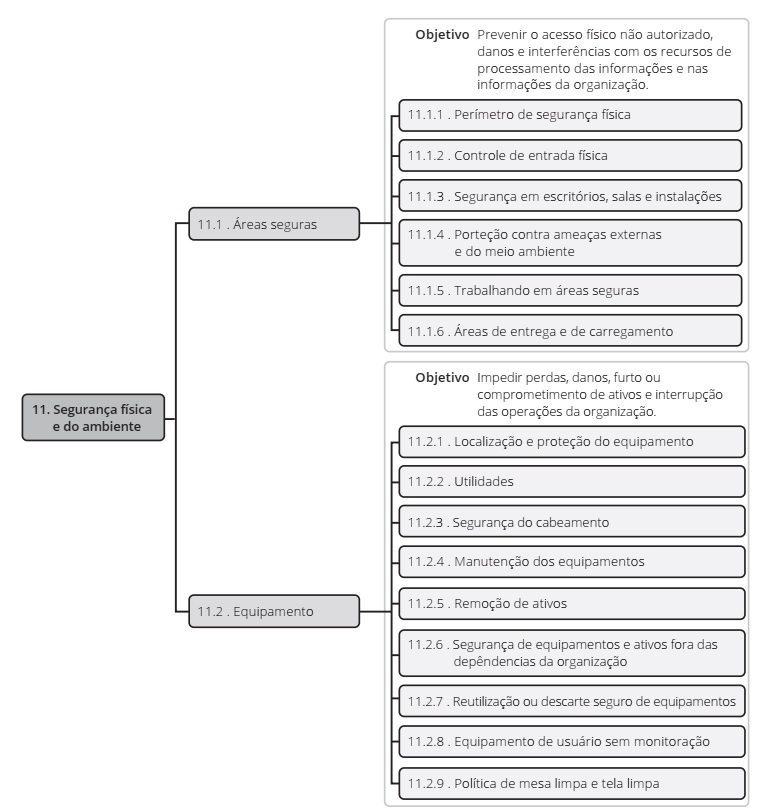
A figura 1 apresenta um resumo da seção 9 da ISO, que demonstra um padrão de boas práticas em segurança informação e as formas de controle; essa ISO detalha cada forma e explica o objetivo da seção.

Partindo desta informação explicaremos que o objetivo da política de controle de acesso é limitar o acesso à informação, recursos e ativos da empresa. Sabendo disso exemplifico que controle de acesso no projeto atuará para limitar e dar acessos apenas às pessoas permitidas. Será utilizado um sistema para liberação de acesso dos usuários e o seu gerenciamento. Utilizaremos um sistema embarcado num minicomputador que será configurado e criado sistema para registrar e liberar o acesso dos usuários. Alguns pontos serão analisados:

* Requisitos de Negócio para criar a medida de controle
* As responsabilidades dos usuários e administradores do sistema.
* A aplicação será com base nesses requisitos e criado o controle de acesso.

Sabendo dessas informações iremos apresentar uma figura que irá resumir outra sessão importante da ISO/IEC 27002. Essa Sessão é a 11 que trata de segurança física e do ambiente.

Figura 2. Seção 11 da ISO 27002:2013



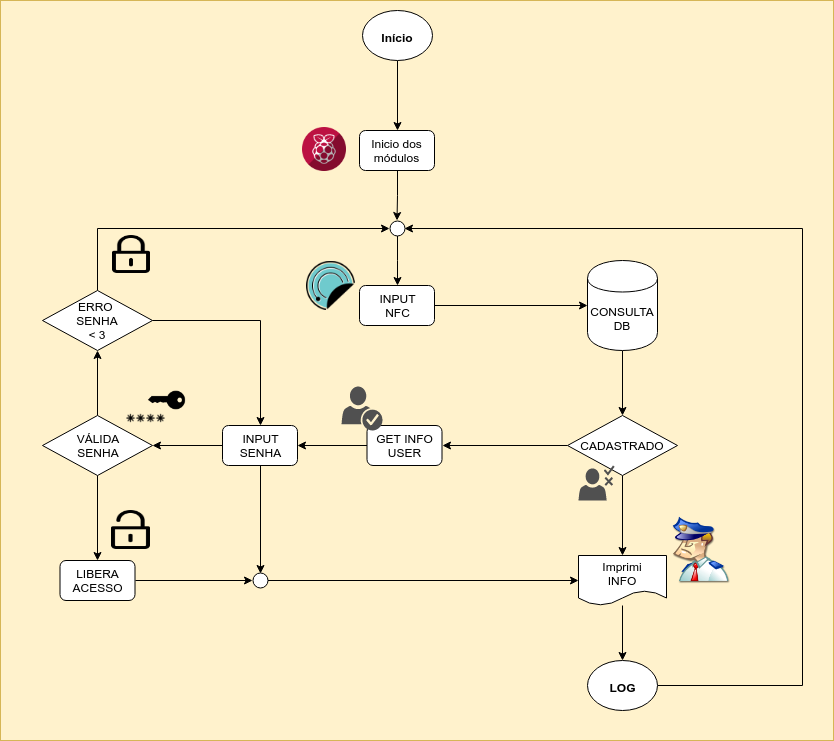
Fonte: Flavia Coelho 2014

Na figura2, a seção 11 é muito mais abrangente, mas iremos adaptá-la para explicar o gerenciamento de acesso e criaremos testes dentro de um ambiente real. Abordaremos o processo de validação e as medidas de segurança física. Serão abordados pontos como:

* O controle de entrada na área segura
* As medidas de proteção na área segura.
* Segurança do equipamento físico e sua utilidade.
* As melhores práticas de manutenção e atualização do sistema.
* As práticas de utiliza e recadastramentos das Tags NFC e sua forma de descarte e atualização.

**1.1 O sistema lógico do controle de acesso.**

A ideia do projeto é criar um sistema embarcado numa placa Raspberry PI e validado o acesso num módulo NFC. O projeto fará integração de duas tecnologias a placa Raspberry PI e as tags do NFC (*Near Field Communication*). A ideia é criar sistema que fique conectado a uma pequena rede de computadores e que serão implementados, medidas de segurança para proteção do sistema. Abaixo um Fluxograma do Sistema.

Figura 3: Fluxograma do Sistema

Fonte: Imagem de nossa autoria

O sistema irá fazer uma captura dos dados contidas nas Tags NFC e armazenado num Buffer temporário e depois disso será feito uma busca no banco de dados para verificar se está liberado o acesso. O sistema fará uma coleta dados através das identidades contidas nas tags NFC e fará uma busca, depois de validado pelo sistema fará uma busca no banco de dados, encontrado o usuário no banco de dados o usuário digitará sua senha pessoal de acesso. Depois de validado a senha o usuário será liberado e passará pela catraca caso contrário terá de digitar a senha novamente ou repetir o processo.

Uma das medidas de controle que adotaremos como padrão é permitir que o usuário digite a senha por três vezes antes de realiza o bloqueio do acesso. O sistema de controle de acesso trabalhará com as seguintes linguagens de programação:

* Sistema de Interface gráfica em HTML com Servidor Web
* Scripts em Python e Shell Linux
* Sistema de captura do NFC por scripts em Shell Linux
* Banco de Dados MySQL

Abaixo uma imagem e primeira fase de desenvolvimento do sistema

Figura 4: Primeira Imagem do Sistema Lógico.



Fonte: Imagem de nossa autoria.

O sistema de controle de acesso irá trabalhar com vários scripts para fazer a comunicação entre a parte física e parte lógica. Criamos a parte lógica em scripts no Linux que irão facilitar essa comunicação entre a parte física.

Scripts em Shell fazem essa comunicação por conseguirem ir a um nível mais baixo conversando diretamente com o hardware através do Kernel. Para Neves (2010) “Um programa em Shell, é uma ferramenta fácil de usar para construir aplicações colando e aglutinando chamadas de sistema, ferramentas, utilitários e compilados”. O programa em Shell é também chamado de script. O script permitirá que o kernel do sistema do Raspberry faça contato entre a parte física e lógica com outros módulos

Fazendo um resumo da parte física.

O Hardware irá possuir:

1. Luzes indicativas (Vermelho/Verde)
2. Teclado NumPad USB
3. Display Lcd 2x18
4. Raspberry Pi B+ (Raspbian Lite)
5. Buzzer
6. Leitor de NFC
7. Cartão com chip NFC
8. Conectores PoE ( Power Over Ethernet) para ligação de enegia ao equipamento

Resumo da utilidade do Hardware no projeto:

* **Luzes indicativas (Vermelho/Verde):** As luzes indicativas serão colocadas para mostrar se irá permitir ou bloquear o acesso. O vermelho mostra sinal padrão de bloqueio ou que hardware vai esperar a sinalização do cartão, o verde sinaliza que será permitido o acesso.
* **Teclado NumPad USB:** O teclado NumPad será utilizado para que o usuário digite a senha.
* **Display Lcd 2x18:** O display será usado para apresentar informações do sistema.
* **Raspberry PI B+:** O Raspberry PI é placa que será utilizada para embarcar o sistema e fazer as configurações do projeto.
* **Buzzer:** O Buzzer será utilizado para dar som ao equipamento e fazer um bip se está liberado ou mais bips se der sinal de bloqueio.
* **Leitor NFC:** O leitor NFC será utilizado fazer a leitura do cartão e validar o acesso.
* **Cartão com chip NFC:** cada usuário terá um cartão desse para utilizar e liberar o acesso da catraca.
* **Conectores PoE ( Power Over Ethernet):** Cada equipamento terá ligado a ele dois adaptadores de energia PoE para ligação do equipamento na Rede Elétrica através do cabo de rede para podermos ter uma maior extensão de ligação do equipamento e também

1. **O que é a placa Raspberry Pi e quais módulos serão utilizados no projeto.**

O Raspberry PI começou com uma ideia de grupo de pessoas que queria criar um computador em uma única placa e que fosse completo. Partindo desse conceito foi criado o equipamento chamado de Raspberry PI.

Um dos seus idealizadores é Eben Upton que criou e fundou a Raspberry PI Foundation. Uma fundação que criou e gerenciou esse equipamento e hoje trata ser canal de comunicação de projetos com Raspberry PI. Uma das empresas que vende o equipamento e ajuda a gerenciar projetos é a Element14 uma empresa americana que atualmente vende a placa Raspberry PI e seus módulos.

Segundo especificações da Raspberry PI o equipamento trabalha com processadores multimídia da Broadcom SoC (system-on-chip em sistema em um chip), Isso significa que a maioria de seus componentes, e incluindo sua unidade central, unidade de processamento gráfico, assim como o hardware de áudio, comunicação, está adaptada e através de um único componente, oculto abaixo do chip de memória, localizado no centro da placa de circuito interno.

O chip BCM2835 trabalha com uma arquitetura de conjunto de instruções diferente o conjunto conhecido como ISA (Instruction set architeture) e essa tecnologia também conhecida como ARM e tecnologia que hoje em são muito utilizados em vários equipamentos mobile. Um resumo do ARM:

A tecnologia ARM combina um conjunto de instruções reduzidas com o baixo consumo de energia. “A arquitetura ARM refere-se a uma arquitetura de processador que evoluiu dos princípios de projeto RISC e é usado em sistemas embarcados” Stallings (2010).

Essa tecnologia foi desenvolvida com necessidade de ter processador que pudesse ter um bom desempenho e um baixo consumo de energia e pudesse ter instruções diretas de hardware e assim facilitando a comunicação entre hardware e o processador. Isso ajudou na criação e implementação dos sistemas embarcados.

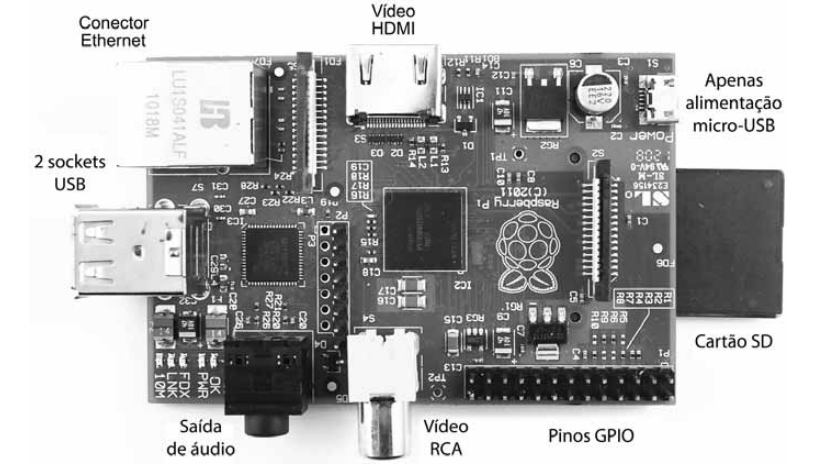
Segundo Willian Stallings uma definição de sistemas embarcados:

Sistema embarcado é uma combinação de hardware e software de computador, e talvez partes adicionais mecânicas e outras, projetada para realizar uma função dedicada. Em muitos casos, os sistemas embarcados fazem parte de um sistema ou produto maior, assim como no caso de um sistema de freios ABS em um carro. (STALLINGS, 2010, p. 35)

Como já foi citada a ideia é criar um sistema de controle de acesso embarcado direto no hardware, isso facilitará e tornar mais rápida a comunicação entre o hardware e o software e dará mais rapidez neste processo.

Escolhemos o Raspberry PI pelo baixo custo e pela capacidade que uma tecnologia dessas pode oferecer ao projeto. Essa tecnologia facilitara criarmos um sistema simples e que poderemos implantar as boas práticas de segurança informação no projeto. A placa Raspberry PI possui vários módulos que podem ser integrados ao projeto. Abaixo uma imagem da anatomia da Raspberry PI:

Figura 5: Anatomia de Raspberry PI modelo B.



Fonte: Symon Monk 2013

Como podemos perceber o Raspberry PI foi criado para ser adaptado e tendo muito de seus componentes integrado a placa, mas podendo ser moldado a necessidade do projeto. Essa placa é uma Placa do Raspberry PI do modelo que apresenta apenas algumas diferenças como memória de 1GB de RAM e tem integrado uma placa de rede wireless. Abaixo apresento o quadro 1 demostra as especificações técnicas dos modelos da Raspberry PI 3 e 2 e Zero.

Quadro 1: Especificações Placa Raspberry PI Zero

|  |  |
| --- | --- |
| Raspberry Pi Zero: Especificações Técnicas | |
| **CPU:** | Broadcom BCM2835, which can run at up to 1GHz. |
| **Ram:** | 512MB |
| **Power:** | 5V, supplied via micro USB connector, drawing 160mA (even when connected to an HD display). |
| **Dimensions:** | 65mm x 30mm x 5mm |
| **Video & Audio:** | 1080P HD video output. Audio output via mini-HDMI connector. |
| **Storage:** | MicroSD card. |
| **Operating System:** | Linux, installed via NOOBS |

Fonte: Element 14 (2019)

A Raspberry Pi Zero é uma placa de baixo custo e tamanho reduzido isso permite que possa ser criado pequenos projetos e de forma compacta, sem perder a flexibilidade que algumas placas da Raspberry proporcionaram.

Quadro 2: Configuração do Raspberry PI 2

|  |  |  |
| --- | --- | --- |
|  | **Raspberry Pi 2 Model B 1GB** | **Model B+** |
| **Processor Chipset** | Broadcom BCM2836 ARMv7 Quad Core Processor powered Single Board Computer running at 900 MHz | Broadcom BCM2835 ARMv6 SoC full HD multimedia applications processor |
| **RAM** | 1GB SDRAM @ 450 MHz | 512 MB SDRAM @ 400 MHz |
| **Storage** | MicroSD | MicroSD |
| **USB 2.0** | 4x USB Ports | 4x USB Ports |
| **Power Draw / voltage** | 1.8A @ 5V | 1.8A @ 5V |
| **GPIO** | 40 pin | 40 pin |
| **Ethernet Port** | Yes | Yes |

Fonte: Element14 (2019)

O quadro 2 já apresenta o modelo do Raspberry Pi 2 que apresenta um aspecto mais robusto e que pode ser utilizado em projetos que necessitam demais uso do hardware. Está será uma das placas que será utilizado no projeto.

Quadro 3: Especificações Técnicas do Raspberry PI

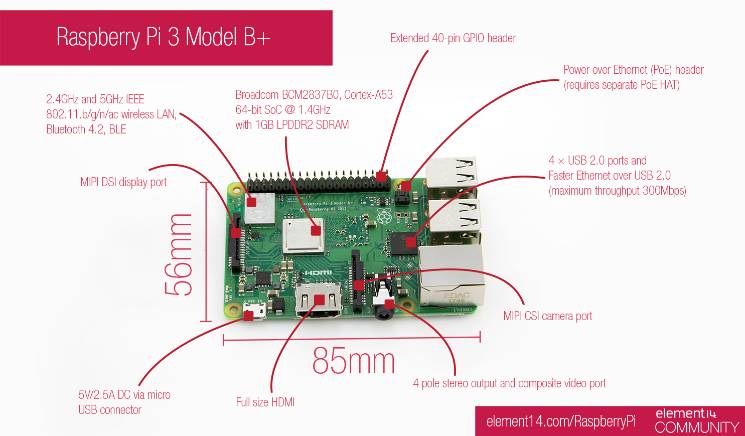
|  |  |
| --- | --- |
| Raspberry Pi 3: Especificações Técnicas | |
| **CPU:** | Broadcom BCM2835, Cortex – A5 64 Bits e SoC 1.4 GHz |
| Ram: | 1GB |
| **Power:** | 5V, drawing 160mA |
| **Dimensões:** | 85mm x 56mm |
| **Vídeo & Áudio:** | HDMI com alta resolução e . Áudio output via microusb conector. |
| **Armazenamento:** | MicroSD card. |
| **Portas USB** | 4 Portas USB |
| **Placa de Rede Ethernet** | Placa Onboard Ethernet com velocidade de 300Mbs |
| **Placa de Rede Wirelles** | Possui uma placa de rede de 2.4 Ghz e conexão com 5GHz e Placa no padrão 802 B/G/N/Ac e uma outra placa para a integração via Bluethoot |
| **Placa GPIO** | Possui placa GPIO de 40 Pinos. |
| **Operating System:** | Linux, instalado via NOOBS |
| **Tecnologia de comunicação** | Possui nova extensão de tecnologia PoE que permite a comunicação de energia através de uma placa de Rede |

Fonte: Element14 (2019)

O Quadro 3 apresenta a informações técnicas da Raspberry 3, esta atualmente segundo fonte do site Element14, é uma das placas mais requisitadas para projetos e através dela podemos criar e aprimorar projetos e fazer conexões com outros equipamentos e inclusive projetos com Arduino.

Na figura 5 apresento uma imagem com foto da placa Raspberry onde apresenta informações de onde fica cada conector e detalhes do Hardware, como posicionamento da placa de rede, conectores GPIO, conector Power Over Ethernet e entre outros.

Figura 5: Especificações Técnicas



Fonte: Element14 (2019)

Uma coisa que é importante citar é os pinos GPIO, eles permitem que o Raspberry Pi se conecte a outros componentes eletrônicos como Arduíno e outras placas microcontroladoras.

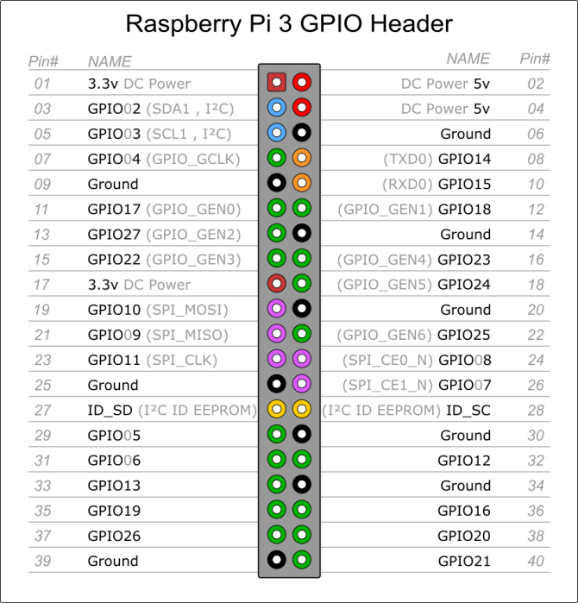
A placa Raspberry Pi Permite, assim que ligue sensores, displays e outros componentes utilizando o conector GPIO de 40 pinos.

GPIO significa General Purpose Input/Output, ou Entrada e saída de uso geral, em tradução livre.

A linha de pinos GPIO é um excelente recurso que fica na borda superior da placa. Um conector GPIO é encontrado em todas as placas Raspberry Pi. Qualquer um dos pinos GPIO pode ser programado e configurado como um pino de entrada ou saída e usado para uma ampla gama de propósitos. RASPBERRY PI FOUNDATION (2019)

Abaixo apresento uma imagem do GPIO.

Figura 6: Cabeçalho do GPIO do Raspberry PI 3.



Fonte: Primeiro Passo Raspberry PI- Filipi Flop.

Abaixo mostro uma imagem do projeto com o NFC integrado na placa.

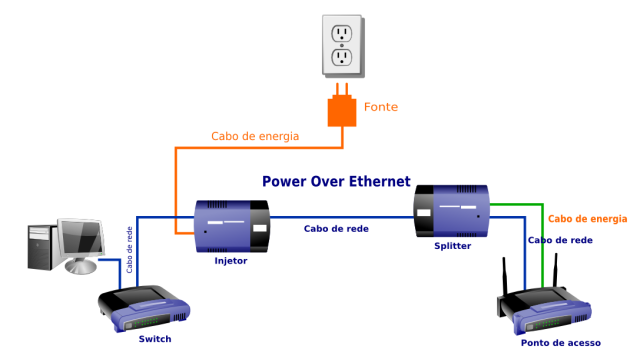
Fonte: Fotos de Montagem do Projeto.

1. **O PoE a fonte de energia do projeto.**

Pensando em algumas situações na hora de realizar a implantação do projeto, tivemos a ideia de permitir criar uma maneira de interligar nosso equipamento de forma rápida e pratica e segura, e tivemos a ideia de implementar esta tecnologia PoE ao projeto.

O Power over Ethernet, ou PoE, é um padrão que permite transmitir energia elétrica usando o próprio cabo de rede, juntamente com os dados, solucionando o problema. Marimoto (2010). Abaixo uma imagem da ideia apresentada pela conexão do PoE:

Figura 7: Modelo de conexão do PoE.



Fonte: Carlos Marimoto (2010)

Para funcionar essa tecnologia necessita que tenhamos um injetor e spliter para realizar a conexão e a ligação do equipamento na energia através da rede, pois necessitamos de apenas umas da pontas esteja ligada a uma tomada e esse equipamento fiquem conectados através de um cabo de rede comum, podemos utilizar para ligar um equipamento na rede de computadores através dele. Abaixo na Figura 8 esta uma imagem de injetor de PoE

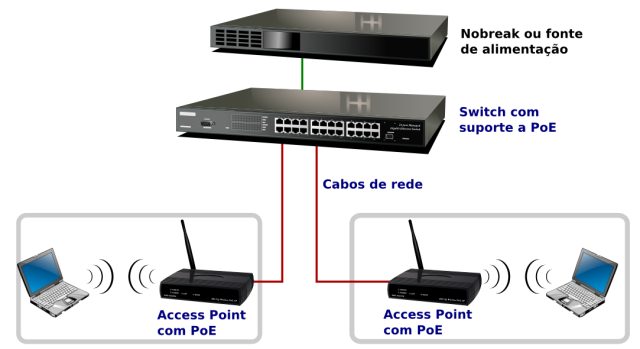
Figura 8: Injetor Poe (sem a fonte) e spliter



Fonte: Carlos Marimoto (2010)

Existem duas situações que será explicada a implementação, essa que foi explicada é que utilizar PoE através de um injetor e um spliter e a outra é atraves de switch PoE que permite interligar vários equipamentos nesse dispositivo e elimina a possibilidade de usar injetores. Abaixo na figura 9 apresentamos um modelo de conexão de switch com suporte de PoE.

Figura 9: modelo de conexão do switch com suporte a PoE.



Fonte: Carlos Marimoto (2010)

Foi criado um padrão de tecnologia para implementação deste dispositivo a IEEE 802.3af. Este padrão foi trabalhado na ideia de pode transformar a mesma corrente contínua que liga os equipamentos de rede poder ligar computadores e outros equipamentos como câmeras, telefones e equipamentos de rede sem fio. Abaixo Citamos algumas vantagens segundo site DIPOL:

**As vantagens desta tecnologia:**

* Transmissão de energia e de dados através de um único cabo isso irá reduzir os custos de instalação e fiação
* Pode chegar até ao alcance de 100 m de cabeamento isso se tiver o cabeamento apropriado)
* Pode trazer mais segurança e confiabilidade e reduzir o risco de choque elétrico por trabalhar com equipamentos de baixa tensão e que é inferior a 60 V) e essa tecnologia pode envolver procedimentos de auto-teste
* Essa tecnologia é fácil de ser instalada e implementada

Como nossos equipamentos trabalha com baixa tensão pensamos em adaptar isso ao nosso projeto, até para poder facilitar e reduzir os custos na hora de implementar e podemos adequar nossa tecnologia para as pequenas empresas.

1. **As aplicações e Ferramentas que serão utilizadas no projeto**

Partindo da premissa de criar todo projeto sistema Linux e utilizando como uma boa base de alguns softwares livres para adaptar e implementá-lo ao projeto. Partiremos da ideia de aprimorar conceitos e utilizar ferramentas como o MariaDb, Aplicações de Linguagem em Python e Scripts em Shell Linux, ferramentas de gerenciamento visual como Bootstrap e sistema operacional embarcado no Raspberry.

A ideia foi criar todo sistema controle de acesso em único equipamento e restringindo e garantindo mais segurança no acesso ao banco de dados que terá sua aplicação conectado de forma externa. As aplicações terão integração e nos facilitará na questão da utilização e do gerenciamento da aplicação.

O software livre que iremos utiliza trabalha com licença de software a GPL ou General Public License (Licença Pública Geral) da Free Software Foundation (FSF) de Richard Stallman, esta é a principal licença do Linux e de muitos outros softwares Open Source. GPL nada mais é que uma designação da licença de software que dá a liberdade aos usuários, dando a todos o direito de usar, compartilhar, estudar e melhorar o software para qualquer finalidade, “ser livre”.

Livre em “software livre” se refere à liberdade, não ao preço. Você pode pagar ou não para obter software do Projeto GNU (Caso você redistribuir software do Projeto GNU, você pode cobrar uma taxa pelo ato físico de transferir uma cópia, ou você pode simplesmente dar cópias de graça.). Sistema Operacional GNU é um sistema de software livre completo, voltado às mais diversas tarefas e compatível com o Unix.

Stallman originalmente criou a licença para o Projeto GNU de acordo com as definições de software livre da Free Software Foundation. Uma vez que você tenha o software, você tem quatro liberdades específicas ao usá-lo:

1. A liberdade de executar o programa, para qualquer propósito (liberdade nº 0)
2. A liberdade de estudar como o programa funciona e adaptá-lo às suas necessidades (liberdade nº 1). O acesso ao código-fonte é um pré-requisito para esta liberdade.
3. A liberdade de redistribuir cópias de modo que você possa ajudar ao seu próximo (liberdade nº 2).
4. A liberdade de aperfeiçoar o programa e liberar os seus aperfeiçoamentos, de modo que toda a comunidade beneficie deles (liberdade nº 3). O acesso ao código-fonte é um pré-requisito para esta liberdade.

Em uma entrevista Stallman cita que em programas privados, repassar cópias é considerado ilegal, já que isso viola os Termos de Uso e Serviço do produto. Mas, entre cometer esse pequeno crime e negar o pedido de um amigo, ele escolhe a primeira opção. “Nesse caso, pelo menos o mal é para quem merece”, argumenta. E, assim como a heresia de falar “sistema operacional [Linux](https://www.tecmundo.com.br/linux)” na frente dele (tem que ser GNU/Linux), outro problema está em se referir ao software livre como open source (ou “de código aberto”).problema está em se referir ao software livre como open source (ou “de código aberto”) (STALLMAN, 2012)

Por curiosidade a palavra GNU significa “GNU's Not Unix” (GNU Não é Unix). Stallman fez o Anúncio Inicial do Projeto em setembro de 1983. Uma versão mais longa, chamada de **Manifesto GNU**, foi publicada em março de 1985. A figura 10, refere-se ao logotipo do Projeto GNU.

Figura 10: Logotipo GNU Wikipedia



Fonte: GNU Wikipedia (2019)

**4.1 Bourne Again Shell – A ferramenta de gerenciamento de script do Sistema Operacional do Nosso Projeto**

BASH "Bourne-Again SHell" ou Shell, é um interpretador de comandos a nível de hardware, ele permite a execução de sequência de comandos inseridos diretamente na linha de comando CLI, PROMPT ou ainda podendo ser lidos de arquivos de texto conhecidos como Shell Script. Esses códigos criados facilitam muito a vida do programador economizando tempo automatizando tarefas e rotinas.

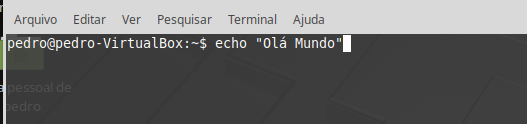
Shell Script é usado em diversos sistemas operacionais, com diferentes dialetos, dependendo do interpretador de comandos utilizado. Um exemplo de interpretador de comandos é o **BASH** que utilizamos em nosso projeto, ele é usado na grande maioria das distribuições **GNU/Linux**. Sua programação é de fácil aprendizagem. Poderíamos ter utilizado outras linguagens para desenvolver o código, mas, como não somos programadores e sim, da Área de Segurança, procuramos minimizar o esforço em programação e dar ênfase aos recursos da que já são oferecidos pelo sistema operacional priorizando o menor custo, que seja simples e que minimize as vulnerabilidades.

Nosso primeiro passo é, saber o que se desejamos fazer, então ver qual o código que executa este comando em shell e aí criar, basta escrever o código em algum editor de texto e salvar. Os comandos podem ser enviados de duas maneiras para o interpretador:

* **Interativa**: Digitados no teclado pelo um a um usuário. Neste modo o computador depende do usuário para executar uma tarefa ou o próximo comando.
* **Não-interativa**: Shell Scripts, neste modo, o computador executa os comandos do arquivo um por um, e dependendo do término do comando, o script pode checar qual será próximo comando que será executado e dar continuidade ao processamento.

De forma interativa forma utilizamos a CLI (Linha de comando), para obter uma resposta em tela. Por mais simples que isso pareça, é assim que se começa uma programação. Abrimos o Shell e digitamos:

Figura 11: Imita o Bash no Linux



Fonte: Imagem de nossa autoria´

Apertamos a tecla ENTER e logo na sequência, obtivemos o seguinte resultado:

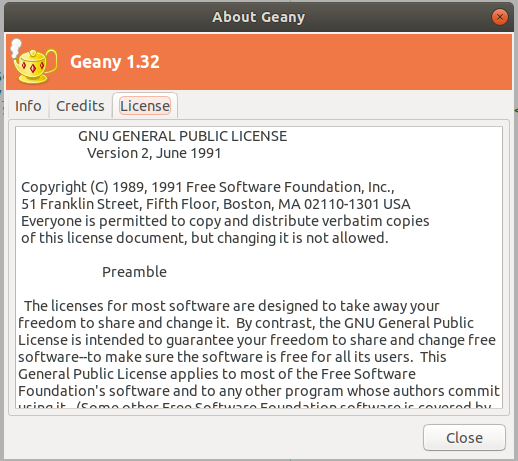
Figura 12: Imita a saída do comando no Bash no Linux



Fonte: Imagem de nossa autoria

Agora iremos utilizar o mesmo comando dentro de um arquivo de texto, para criar o nosso primeiro Script Shell, de forma não interativa, para isso precisamos abrir um editor de texto de nossa preferência.

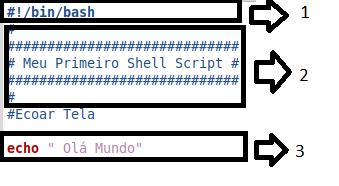
Figura 13: Imagem Geany 1.32



Fonte: Imagem de nossa autoria

Neste caso utilizaremos o **Geany 1.32,** assim visto na figura 13. Que é um editor de texto avançado que associa muitas linguagens em uma plataforma única para uma fácil utilização. Podemos observar nos créditos que o Geany também é baseado na licença GNU.

Figura 14: Modelo Script



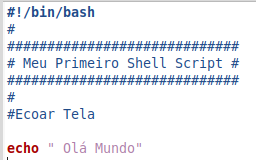
Fonte: Imagem de nossa autoria

A Figura 14 mostra um arquivo de texto **ola\_mundo.sh,** contendo 8 linhas escritas, nas quais existem 3 itens cada qual com sua funcionalidade, são eles:

1. Linha 1: SheBang, refere-se aos dois primeiros caracteres que aprontam qual será o interpretador que irá executar as tarefas dentro do arquivo.
2. Linhas 3 - 5: Comentários, geralmente iniciado com #, são linhas que utilizamos para comentar algo que posteriormente possa ser revisado de forma mais clara, não precisando traduzir o programa ou script.
3. Linha 8: Comando a ser executado. Neste caso estamos efetuando um comando para exibir na tela Olá Mundo!!!

Em resumo nosso Shell Script, se resume em 2 linhas, uma que mostra ao Shell qual será o interpretador, neste caso o Bash e o comando a ser executado.

Figura: 15 conteúdo do arquivo de texto ola\_mundo.sh



Fonte: Imagem de nossa autoria

Nosso pequeno script cresceu 6 linhas que aparentemente não precisamos, mas acredite, é uma ótima prática em programação criar comentários para que outros possam ler o que foi transcrito em linguagem de programação., de forma simples que não faça perder tempo. Vale lembrar que foi escrito por nós embora sejam comando conhecidos as coisas podem se tornam mais complexas futuramente, daí um bom comentário fará do nosso script amador em um script de qualidade.

Um bom profissional sabe como fazer um trabalho de qualidade. Isso faz parte do seu dia a dia, não é preciso se esforçar. A atenção aos detalhes e a busca por um resultado perfeito são algumas de suas características. Aprenda quais são os componentes que farão seu programa em Shell ser considerado profissional. Destaque-se faça o melhor, faça o todo”. (Aurélio Marinho, 2011, p. 27)

Com nosso script pronto agora vamos a parte de execução do nosso “Arquivo de Texto” ola\_mundo.sh.

Figura 16: simula execução de um arquivo via linha de comando.



Fonte: Imagem de nossa autoria

**Permissão Negada** como assim permissão negada? Por um simples motivo, algumas linhas acima citamos que poderia ser interpretado pelo Bash, mas não foi. Estranho, não? Não! O que aconteceu é que no nosso caso em Linux todo arquivo pode ser interpretado de 3 maneiras RWX:

* R **Leitura**: é permitido ler
* W **Escrita**: é permitido escrever
* X **Execução**: é permitido executar

O quadro 4 exibe os atributos das permissões e seu respectivo valor numérico

|  |  |
| --- | --- |
| Tipos de Permissão | Modelos de Acesso |
|  |  |

Quadro 4 : Emerson Alecrin 2010

Cada uma dessas letras carrega um valor numérico que nos diz o que é possível se fazer com tals arquivo. Ainda quando se trata de permissões existem 3 subdivisões que possuem as permissões U. G. O :

* User: usuário especifica o dono do arquivo
* Group: específica a qual grupo pertence
* Other: especifica qual permissão terceiros podem ter no arquivo

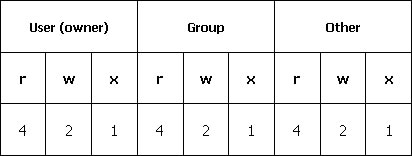


Figura 12: PPLWARE - Linux-permissões-em-ficheiros-e-diretórios

Sabendo como trabalhar com permissões podemos aumentar o nosso nível de segurança em nosso Shell Script. Com base neste conteúdo vamos analisar porque nosso programa não foi executado, na figura 12 vamos verificar suas permissões e alterar para que fique um “Arquivo de Texto” executável.

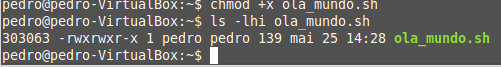
Figura 17: lista e chega atributos



Fonte: Imagem de nossa autoria

Notasse que o arquivo exibe as permissões (-rw-r-r--), não existe nenhuma permissão de execução somente leitura e escrita. Para que não percamos o foco principal do trabalho, não iremos nos aprofundar muito neste tema e por se tratar de um conteúdo muito amplo e que demanda um estudo mais aprofundado. Para resolver nosso problema iremos alterar com o seguinte comando:

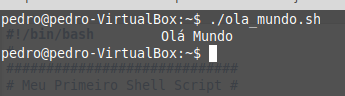
Figura 18: alterando os atributos do arquivo



Fonte: Imagem de nossa autoria

A Figura 18 nos retorna os novos atributos do nosso arquivo (-rwxrwxr-x), tornando-o não mais apenas um “Arquivo de texto”, e sim um arquivo executável.

Figura 19: executando o Shell Script



Fonte: Imagem de nossa autoria

Na Figura 19, podemos executar nosso arquivo e obter o resultado esperado nosso **Olá Mundo!!!**.

Para complementar e entender como funcionar como foi alterado a permissão no Quadro 6, seguem 2 tabelas com mais detalhes de permissionamento e os seus respectivos atributos que contém em arquivos e diretórios de Linux. Para atribuir permissões a diretórios é necessário utilizar a seguinte sintaxe: chmod –R

|  |  |
| --- | --- |
| Tabela de Permissões de Acesso de Usuário e diretórios | Modelo de Permissões de Acesso de usuário e diretório |
|  |  |

Quadro 7 : Pedro Pinto (2006)

Para atribuir permissões a diretórios é necessário utilizar a seguinte sintaxe: chmod –R. Desta forma podemos começar a produzir os nossos scripts, para no final do projeto o transformarmos em um programa que irá gerenciar nossa aplicação.

* 1. **GIT e GITHUB**

Git é um sistema de controle de versão de arquivos que armazena localmente todo conteúdo versionando a cada modificação. Nele é possível desenvolver projetos no qual diversas pessoas podem contribuir simultaneamente no mesmo, editando e criando novos arquivos e permitindo que os mesmos possam existir sem o risco de suas alterações serem sobrescritas. Imagina o caos que seria caso um mesmo código com várias linhas fosse escrito por mais de uma pessoa ao mesmo tempo

Já o GitHub é um serviço de web compartilhado que oferece diversas funcionalidades extras aplicadas ao git e um serviço na nuvem para armazenar os projetos versionados pelo GIT. O Github é similar a uma rede social de programação. Onde tudo que se faz pode ser compartilhado ou não, dependendo do plano que se tem dentro da plataforma.

Usaremos o Github como nosso servidor remoto para o armazenamento do nosso código fonte. Por ser uma plataforma gratuita, simples e de fácil utilização que nos permite um desenvolvimento ágil com mais de uma pessoa envolvida no mesmo projeto, fácil acesso e gera um histórico de modificações caso precisemos voltar em uma versão anterior.

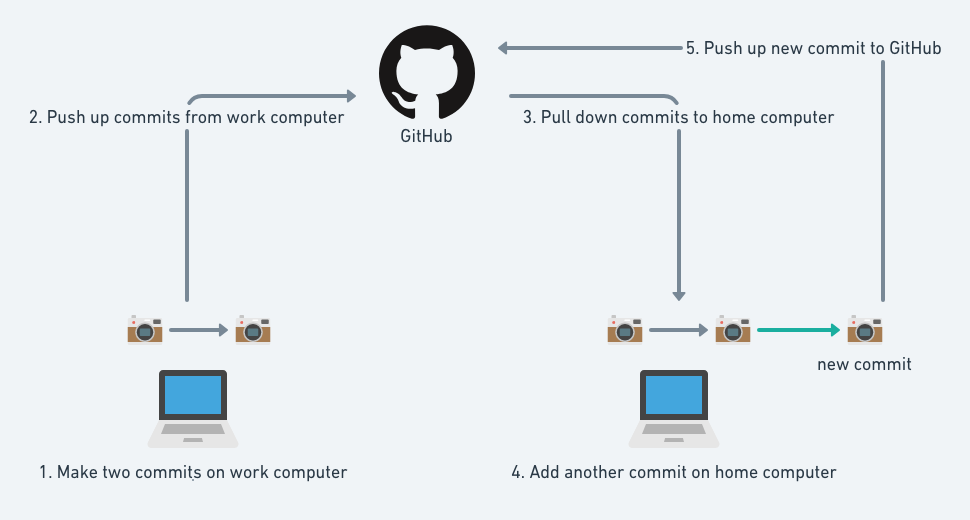


Figura 20: David Whitaker (2018)

A Figura 15, mostra o fluxo de funcionamento do Github. Nela é possível ver cinco etapas nas quais foram feitas cinco modificações de um mesmo arquivo em dois locais diferentes, sempre enviando ao mesmo repositório, assim evitando um conflito no código fonte. Nosso código fonte se encontra dentro do repositório:

**https://github.com/MAUQUINHOS/DEV/NFC\_RPI**

**4.3 Maria DB - Nossa Base de dados**

Em nosso projeto utilizaremos MariaDB, devido ao fato dele ter um monte de novas funcionalidades agradáveis e utilizar a licença GPL, como grande parte de nossas ferramentas. Além de tudo, vai funcionar exatamente igual ao MySQL não há nenhuma necessidade de converter um bancos MySQL de dados para migrar para o MariaDB. Mas não se resume apenas a isso, o sistema pode ser aplicado a outros bancos de dados. O MariaDB é o mesmo que o MySQL, todos os comandos, interfaces, bibliotecas e APIs é um sistema de gerenciamento de banco de dados relacional(SGBD), com um modelo cliente-servidor, que utiliza a linguagem SQL (Linguagem de Consulta Estruturada, do inglês Structured Query Language) como interface.

Suas principais características são:

* Portabilidade;
* Compatibilidade;
* Desempenho e estabilidade;
* Pouco exigente quanto a recursos de hardware;
* Facilidade de uso;
* É um Software Livre com base na GPL;
* Replicação facilmente configurável;
* Interfaces gráficas;

Uma característica fundamental do MariaDB é, ser desenvolvido em código aberto e funcionar em muitos sistemas operacionais, podemos dizer que o MariaDB é um fork isso quer dizer, um projeto baseado no MySQL. Com MariaDB ou MySQL, é possível utilizar ambos em sistemas de produção de alta exigência e em aplicações sofisticadas.

Quando profissionais desenvolvem aplicações "mais complexas" onde há a necessidade de armazenamento de informações que utilizem banco de dados, muitos optam por linguagens como PHP e não Shell Script como no nosso caso, pelo fato do Shell Script não se tratar de um programa Front-end que é responsável por “dar vida” à interface, e sim, a parte da aplicação que interage diretamente com o usuário.

Estes profissionais focam em HTML (linguagem de marcação), JavaScript (linguagem de script/programação), PHP, Java dentre outras que se comunicam com os dados armazenados através do SGDB(Sistema Gerenciador de Banco de Dados). Nestas linguagens, existem frameworks para lidar com manipulação de dados relacionais. Para entender um pouco como funciona isso podemos usar a Figura 21 que mostra a arquitetura de um Sistema Operacional e a Figura 22 que mostra a arquitetura do SBDG.

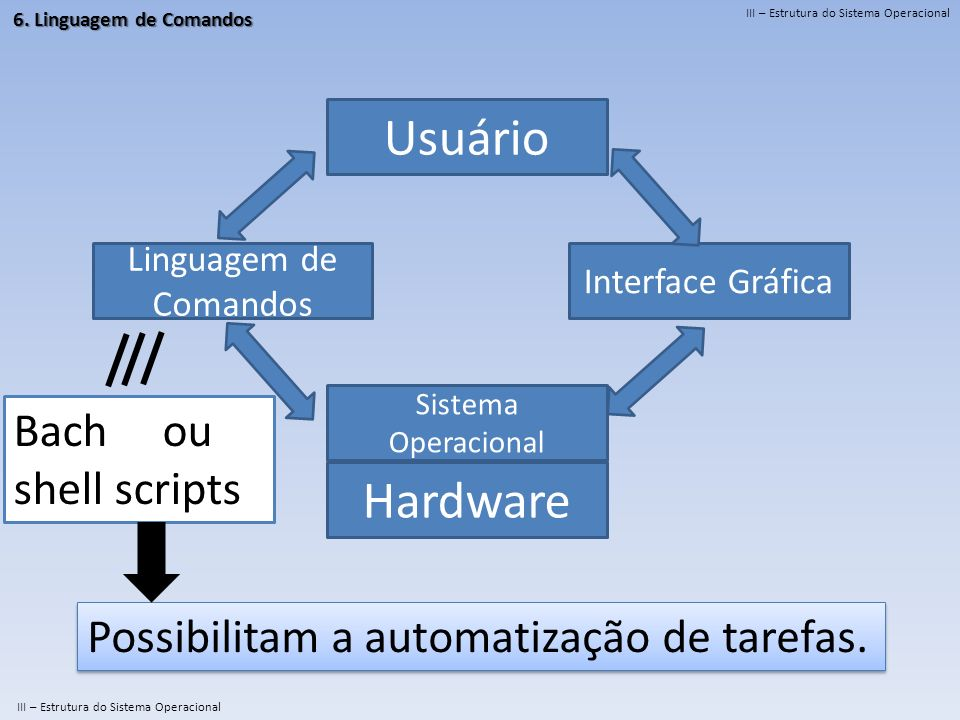


Figura 21: <https://slideplayer.com.br/slide/6856346/>

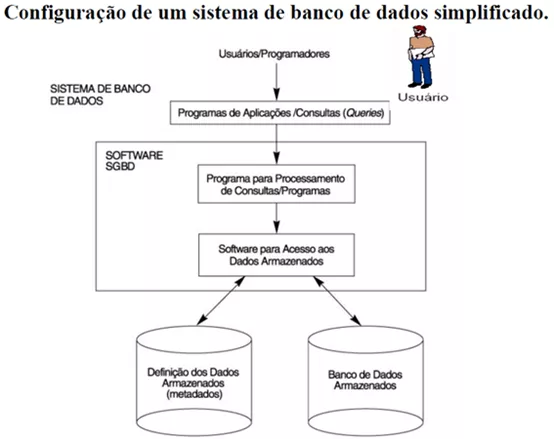


Figura 22 : Cooperarit - banco-de-dados-conceituando-banco-de-dados-e-sgbd

Já na Figura 23 conseguimos observar como é o acesso físico do usuário ao interpretador de comando Bash de forma local e onde fica a aplicação que gera as vulnerabilidades do sistema devido as configurações ou sistema com brechas de segurança.

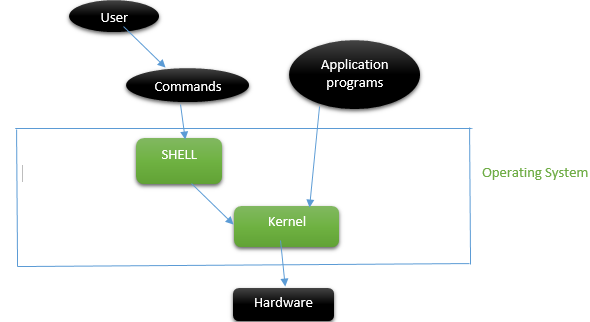


Figura 23: Stackflowover - bash-script-tutorial-write-your-own-terminal-commands

No nosso caso, o uso do shell é muito utilizado para automatização, diferentemente do caso dos programadores em Front-End, este foi um de nossos problemas. Precisamos receber os dados de um Hardware NFC e para isso seria necessário a implementação de frameworks tornando os recursos de nosso equipamento mais limitado, além de adicionar uma camada a mais de insegurança no equipamento, onde temos a visão que muitos dos ataques são provenientes de aplicações mal configurada ou vulneráveis.

Segundo um documento da Owasp gerado anualmente TOP10 OWASP. No ano de 2017, os principais ataques que comprometem o risco de segurança são gerados através de Injeção de comando que é um ataque no qual o objetivo é a execução de comandos arbitrários no sistema operacional do host através de um aplicativo vulnerável.(OWASP, 2017, p. 02)

Pensando nisso teríamos de imaginar, como fazer para tentar trazer uma melhora na usabilidade dos scripts, e além do mais, não deixar permissões excessivas que possam vir a prejudicar o projeto.

Após pesquisarmos muito sobre o assunto, notamos que era possível integrar Shell ao Hardware NFC e também fazer um Script acessar um banco de dados sem precisarmos instalar frameworks ou linguagens adicionais tudo isso já integrado ao nosso sistema, e, ainda de quebra, com mais um script poderíamos criar um mini servidor para que pudéssemos gerenciar tudo através de um local remoto. Assim começamos a desenvolver o projeto que ficou similar ao fluxo da Figura 24.

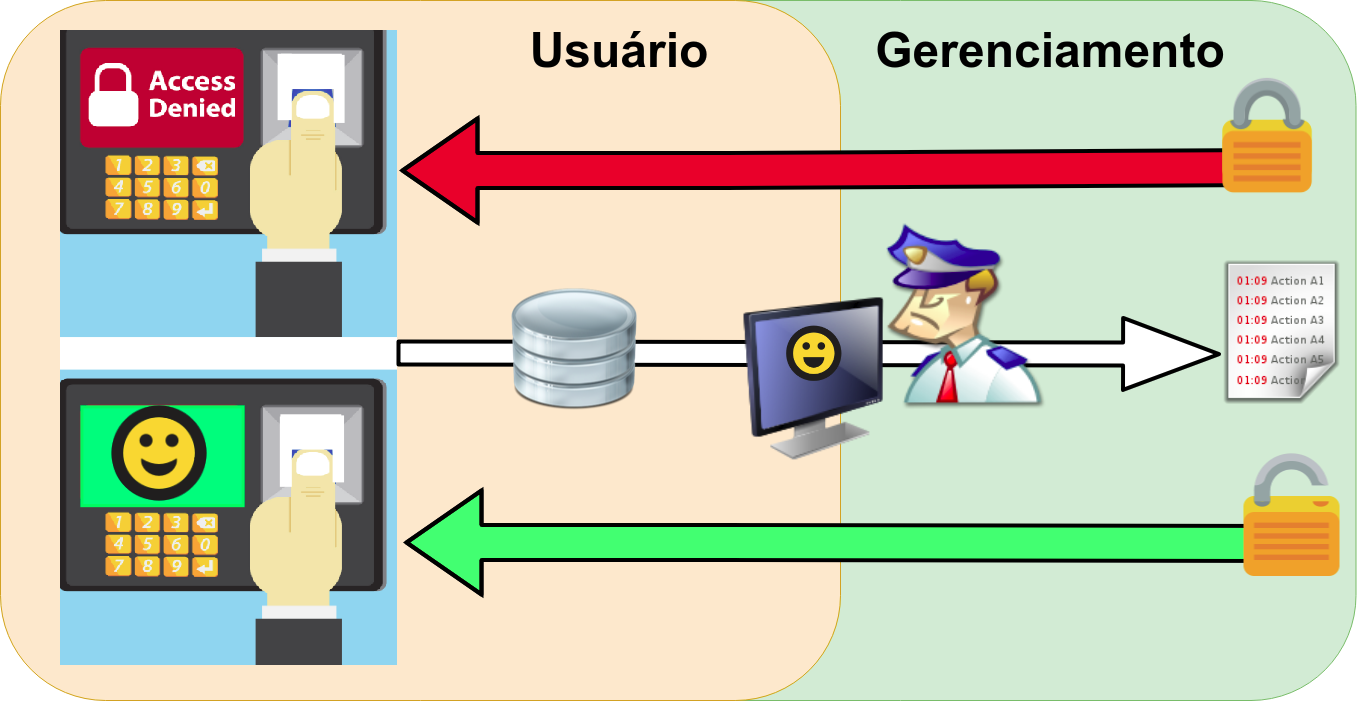


Figura 24: Imagem de nossa autoria

Após concluirmos esse fluxo começamos a nos preocupar como poderíamos aplicar mais segurança, funcionalidade e alta disponibilidade em nosso projeto, mas isso para um próximo tópico onde abordaremos técnicas de hardening e segurança utilizando normas internacionais.

Para finalizarmos sem nenhum tipo de dúvida, nosso sistema aceitará a entrada do usuário remoto, validar o input para que não seja exposto a nenhum tipo de injeção de comandos, irá executar alguns comandos pré-definidos podendo liberar ou não o acesso ao ambiente.

.

**4.4 BootStrap - Gerenciamento de sistema no projeto**

Agora que temos a base de funcionamento de nosso projeto, precisamos saber como gerenciar de forma simplificada a automação e acessar as informações contidas na base de dados. Pensando nisso vimos que a melhor forma de fazer o gerenciamento é a utilização de uma interface via web. Para isso usaremos o framework BootStrap que basicamente é uma ferramenta de desenvolvimento web front-end e que visa facilitar e agilizar o trabalho, oferecendo padrões para HTML, JavaScript e CSS. De fácil implementação trabalha sobre a licença MIT recomendada pela Free Software Foundation a mesma culminou na GPL.

O Bootstrap tem em sua padronização uma facilidade que permite que os sites tenham um melhor aspecto, é muito utilizado na criação de sites responsivos (mobile). Agilizando a produtividade e evitando perder tanto tempo digitando toda uma linha de formatação do html. Contém inúmeras bibliotecas prontas disponíveis, o único trabalho que o desenvolvedor tem é de, praticamente, só as incluir em seus projetos.

Conforme a Figura 25, o gerenciamento pode ser flexível desde um monitor em uma sala, até mesmo de forma móvel em um smartphone ou em um tablet desde que permaneça na mesma infraestrutura de rede em que o sistema esteja implementado.

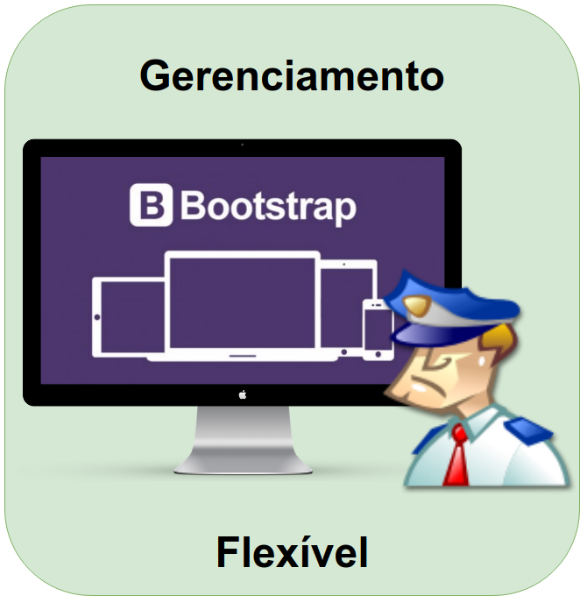


Figura 25: Imagem de nossa Autoria

Agora seguiremos com mais 2 imagens que mostram a interface elaborada e a flexibilidade do Bootstrap para criação de uma aplicação multiplataforma e com aparência de projeto web profissional.

Na Figura 26 observa-se a aplicação rodando em um ambiente com um monitor de menor resolução exemplo 800x600px. Onde no número 1, o menu é transformado em ícones de forma que seja tudo adaptado ao tamanho da tela número 2.



Figura 26: Imagem de Nossa Autoria.

Já na Figura 27, observamos que os ícones exibidos no número 1 aparecem em forma de menus escritos e onde a tela tem uma disposição de área maior número 2, pode-se imaginar que isto seria uma tela do tamanho 1024x768px quase o dobro da tela exibida na Figura 26,  a visualização é facilitada e todo conteúdo é melhor interpretado. Isso nos mostra o quão foi otimizado nosso tempo e conseguimos um resultado muito bom.



Figura 27: Imagem de Nossa Autoria

E com esta interface mais amigável concluímos o que será nosso futuro painel de gerenciamento de nosso projeto.

**5. Python a Linguagem de Dinâmica do Sistema de Controle de Acesso**

Python é uma  linguagem de altíssimo nível (VHLL - Very High Level Language), de sintaxe moderna, orientada a objetos, programação multiplataforma, de tipagem dinâmica e forte, interpretada e interativa.



Figura 28: logotipo oficial Python(Python.org)

Python contém uma licença própria Copyright da PSF (Python Software Foundation) que é compatível com a GPL e contém poucas restrições.

As linguagens dinâmicas eram vistas no passado apenas como linguagens script, usadas para automatizar pequenas tarefas, porém, com o passar do tempo, elas cresceram, amadureceram e conquistaram seu espaço no mercado, a ponto de chamar a atenção dos grandes fornecedores de tecnologia.

Vários fatores contribuíram para esta mudança, tais como a internet, o software de código aberto e as metodologias ágeis de desenvolvimento.

A internet viabilizou o compartilhamento de informações de uma forma sem precedentes na história, que tornou possível o crescimento do software de código aberto. As linguagens dinâmicas geralmente são código aberto e compartilham as mesmas funcionalidades e em alguns casos, os mesmos objetivos. (Luiz Borges, 2010, p:11)

Com esta linguagem não tivemos tanta dificuldade por ela ser muito similar ao Shell Script e conter um vasto conteúdo na internet, ela é denominada como uma linguagem dinâmica baseada em scripts que diferente do BASH utiliza a extensão .py e um framework uma infinidade de bibliotecas moldáveis para diferentes situações. Em nosso caso ela será utilizada como um complemento para nosso pequeno servidor. Por ser bem flexível e fácil integrar com outras linguagens. Este é um resumo básico da Linguagem Python e sua usabilidade em nosso projeto.

**6. O que é o NFC**

Near-Field Communication (NFC) ou, Comunicação por campo de proximidade (CCP), é uma tecnologia que permite a troca de informações sem fio entre dispositivos compatíveis que estejam próximos, a comunicação é estabelecida automaticamente, sem a necessidade de configurações adicionais. Estes dispositivos podem ser telefones celulares, tablets, crachás, cartões de bilhetes eletrônicos, pulseiras e qualquer outro dispositivo que tenha um chip NFC.

Um dos maiores exemplos que podemos ver são os cartões utilizados para validar passagens tanto em metro quanto em ônibus. Esta tecnologia, ela está cada vez mais presente em nosso dia a dia.

Tipos de transmissão

* + Passivo: apenas um dos dispositivos gera o sinal de conexão, o segundo apenas recebe. Assim é possível colocar etiquetas CCP em itens que não recebem alimentação elétrica direta, como cartões, embalagens e cartazes.
  + Ativo: ambos dispositivos geram o sinal, por exemplo, um smartphone e um receptor no caixa de uma loja.
  + Leitura e gravação: leitura ou alteração de dados em um dispositivo CCP, como um receptor que desconta créditos registrados em um cartão de viagens.
  + Peer-to-peer: cada dispositivo pode tanto receber quanto enviar dados para o outro, por exemplo, para a troca de arquivos entre dois celulares.

A Figura 29, mostra a dissolução em acetona de um cartão muito comum utilizado nesta tecnologia NFC.



Figura 29: Gabriel Pato (2018)

Já na Figura 30, logo após dissolver conseguimos observar um microchip com um alguns fios que servem de antena. Sua antena ao se aproximar do receptor ativo torna-se energizada e alimenta o chip.



Figura 30: Gabriel Pato (2018)

Na Figura 31, conseguimos ver um receptor facilmente encontrado na internet. Nele é gerado um campo eletromagnético que energiza a antena, assim alimentando o chip que contém um código em formato Hexadecimal. Esse código é lido pelo receptor e utilizado como um identificador(ID).



Figura 31: Gabriel Pato (2018)

Esta já é uma tecnologia um pouco antiga, ao contrário do que se é pensado. Criada em 1994 pela empresa Mifare e batizado como Mifare Classic, para ser ter uma noção, quando esta tecnologia foi inventada o WIndows ainda era o 3.1 um sistema operacional de 16 bits.

Mesmo havendo várias novas versões disponíveis o Mifare Classic ainda é o mais utilizado no mundo inteiro. Isso não quer dizer que seja de certa forma o mais seguro. Existem duas versões oficiais, a única diferença entre elas é o espaço interno uma contendo 1kb e a outra contendo 4kb. Muito similar a um disco rígido, este cartão é dividido em blocos e setores. Cada bloco contém 16 bytes que em Hexadecimal representa de 0 a F [0123456789ABCDEF] e cada setor contém 4 blocos.

Um cartão de de 1kb tem 16 setores e um cartão de 4kb contém 64 setores.

Não vamos nos atentar muito em em como é feito a gravação deste cartão e sim em específico no setor de número 0. Como podemos ver na Figura 32.

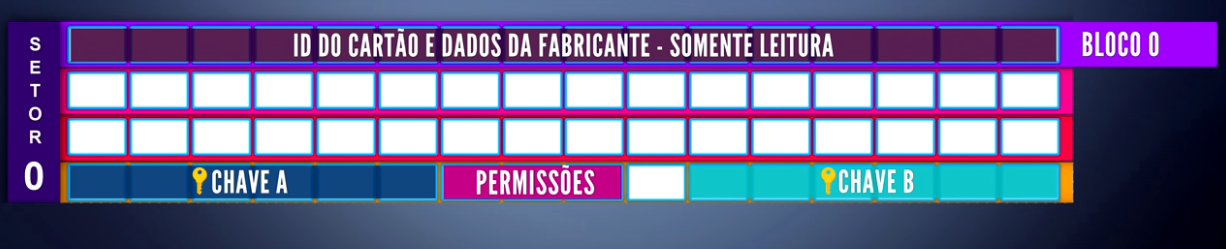


Figura 32: referência youtube

Neste setor de número 0 e bloco 0 é contido o número identificador do cartão que já vem configurado de fábrica, não podendo ser modificado é somente acessado no modo leitura. Isso teoricamente é uma maneira que o Mifare viu de segurança. Mais isso mesmo já foi descartado pela própria empresa em seu site:

Seguindo a ampla aceitação da tecnologia de emissão de bilhetes sem contato e o sucesso incrível da linha de produtos MIFARE Classic, os requisitos de aplicação e necessidades de segurança aumentam constantemente. **“Dessa forma, não recomendamos mais o uso do MIFARE Classic em aplicações nas quais a segurança é uma questão primordial”.**

Como nossa intenção sempre foi utilizar tal tecnologia. Nos colocamos o desafio de como poderíamos criar um produto de baixo custo com uma grande eficiência e um nível maior de segurança. Aceitamos o desafio e começamos a desenvolver .

**Referências**

ABNT – Associação Brasileira de Normas Técnicas. **ABNT NBR ISO/IEC 27001 – Tecnologia da informação – Técnicas de segurança – Sistemas de gestão de segurança da informação – Requisitos**. ABNT, 2013.

ABNT – Associação Brasileira de Normas Técnicas. **ABNT NBR ISO/IEC 27002 – Tecnologia da informação – Técnicas de segurança – Código de prática para a gestão de segurança da informação**. ABNT, 2013.

CHACOS, B. **Raspberry Pi 3 review: The revolutionary $35 mini-PC cures its biggest headaches.** Disponível em: < http://www.pcworld.com/article/3057888/computers/raspberry-pi-3-review-the-revolutionary-35-mini-pc-cures-its-biggest-headaches.html >. Acesso em: 18 de setembro de 2018.

COELHO, F. E. S, ARAUJO, L. G. S, BEZERRA, E. K, **Gestão da segurança da informação:** NBR 27001 e NBR 27002; Rio de Janeiro: RNP/ESR, 2014.

ELEMENT 14, **Raspberry PI 3 Modelo B + Especificações técnicas**. Disponível em < https://www.element14.com/community/docs/DOC-88853/l/raspberry-pi-3-model-b-plus-b-technical-specifications> Acesso em: 06 de novembro 2018

FERNANDES, A. A., **Implantando a governança de TI:** da estratégia à gestão dos processos e serviços 3 ed. Rio de Janeiro: Brasport, 2012.

FLOP, F. **Primeiros passos com Raspberry** PI. Disponível em < https://www.filipeflop.com/blog/primeiros-passos-raspberry-pi-e-linux/> Acesso em: 07 de novembro de 2018.

MONK, S. **Programando o Raspberry Pi** - Primeiros Passos com Python. São Paulo: Editora Novatec, 2013.

NEVES, J. **Programação Shell Linux**. Rio de Janeiro: Editora Brasport, 2010.

RICHARDSON, M. **Primeiros passos com o Raspberry Pi**; tradução Patrícia Tieme Maeda. São Paulo: Novatec Editora, 2013.

RASPBERRY PI FOUNDATION. **Raspberry Pi 3.** Disponível em: https://www.raspberrypi.org/products/raspberry-pi-3-model-b/>. Acesso em: 18 de setembro de 2018

STALLINGS, W. **Arquitetura e Organização de Computadores 8 ed.** Tradução: Daniel Vieira, Ivan Bosnic e Revisão Técnica: Ricardo Pannain. São Paulo: Pearson Pratice Hall, 2010.

UPTON, E. , HALFACREE, G. **Raspberry PI Manual do Usuário.** Tradução Celso Roberto Paschoa. São Paulo: Novatec editora, 2013

BARBIERI, L. **O que é bootstrap e para que serve ?** . Disponível em: < https://www.ciawebsites.com.br/dicas-e-tutoriais/o-que-e-bootstrap/> Acesso em 02 abril de 2019

PATO, G. Cartões de transporte público são hackeáveis? - parte 1: Como Funcionam. (16m46s). Disponível em: <https://www.youtube.com/watch?v=QHcTviUU80A>. Acesso em: 03 maio 2019

WIKIPEDIA, **O que é o NFC – (What is NFC?).** Disponível em: < https://pt.wikipedia.org/wiki/Near\_Field\_Communication > Acesso em 14 de Maio de 2019.

MIFARE; **Linha MIFARE Classic**. Disponível em: < https://www.mifare.net/pt/products/ics-de-cartoes-com-chip/linha-mifare-classic/> Acesso em: 03 maio 2019

ISO/IEC 18092:2004. **Information technology -- Telecommunications and information exchange between systems -- Near Field Communication -- Interface and Protocol (NFCIP-1).** Geneva: [*s. n.*], 2004.

MARIADB. **Sobre o MariaDB**. [*S. l.*]. Disponível em: < https://mariadb.com/kb/pt-br/sobre-o-mariadb/.> Acesso em: 3 maio 2019.

WIKIPEDIA. **MySQL**. Disponível em: < https://pt.wikipedia.org/wiki/MySQL.> Acesso em: 11 maio 2019.

PINTO, P. **6 dicas de segurança para o MySQL/MariaDB**. Disponível em: < https://pplware.sapo.pt/truques-dicas/6-dicas-de-seguranca-para-o-mysqlmariadb/. > Acesso em: 4 maio 2019.

OFICINA DA NET. **MySQL - o que é?**. Disponível em: < https://www.oficinadanet.com.br/artigo/2227/mysql\_-\_o\_que\_e. > Acesso em: 4 maio 2019.

VIANA, D. **O que é front-end e back-end?**. Disponível em: <https://www.treinaweb.com.br/blog/o-que-e-front-end-e-back-end/.> Acesso em: 4 maio 2019.

OWASP. **OWASP Top 10 -2017 The Ten Most Critical Web Application Security Risks.** Disponível em: <https://www.owasp.org/images/7/72/OWASP\_Top\_10-2017\_(en).pdf.pdf.> Acesso em: 5 maio 2019.

OWASP. **About The Open Web Application Security Project**. Disponível em: < https://www.owasp.org/index.php/About\_The\_Open\_Web\_Application\_Security\_Project. > Acesso em: 5 maio 2019.

STALMAN, Richard. **GNU GENERAL PUBLIC LICENSE v3.0.**  Disponível em: < https://bash.cyberciti.biz/guide/GPL. > Acesso em: 04 maio 2019.

WIKIPEDIA. **GNU General Public License**. Disponível em: < https://pt.wikipedia.org/wiki/GNU\_General\_Public\_License.> Acesso em: 4 maio 2019.

WIKIPEDIA. **Projeto GNU**. Disponível em: < https://pt.wikipedia.org/wiki/Projeto\_GNU.> Acesso em: 4 maio 2019.

BERALDO, R. **Visão Geral do Sistema GNU**. Disponível em: < https://www.gnu.org/gnu/gnu-history.pt-br.html.> Acesso em: 4 maio 2019.

LOZANO, F; FONTENELLE, R. **O Manifesto GNU**. Disponível em: <https://www.gnu.org/gnu/manifesto.html.> Acesso em: 4 maio 2019.

BERALDO, R; FONTENELLE, R. **O que é o software livre?**. Disponível em: < https://www.gnu.org/philosophy/free-sw.html.> Acesso em: 4 Abril 2019.

FSFE. **O que é o projeto GNU?**. Disponível em: < https://fsfe.org/freesoftware/basics/gnuproject.pt.html.> Acesso em: 4 abril 2019.

KLEINA, N. **Conversamos com Richard Stallman, o guru do software livre. TECMUNDO,** Disponível em: < https://www.tecmundo.com.br/software-livre/34146-conversamos-com-richard-stallman-o-guru-do-software-livre.htm.> Acesso em: 4 abr. 2019.

WIKIPEDIA. **BASH**. Disponível em: < https://pt.wikipedia.org/wiki/Bash.> Acesso em: 4 abr. 2019.

DELFINO, P. **Programação shell script: O básico para iniciar e criar scripts no bash do Linux**. Disponível em: <https://e-tinet.com/linux/programacao-shell-script/.> Acesso em: 4 abr. 2019.

JARGAS, A. M. **Shell Script Profissional – Aurelio Marinho**. São Paulo: Novatec, 2008.

REUBER, R. **Bash - O interpretador de comandos**. Disponível em: <https://www.vivaolinux.com.br/artigo/Bash-O-interpretador-de-comandos.> Acesso em: 4 abr. 2019.

WENDLING, C. *et al*. **IDE GEANY**. Disponível em: < https://www.geany.org/.> Acesso em: 8 maio 2019.

WIKIPEDIA. **SHEBANG**. Disponível em: < https://pt.wikipedia.org/wiki/Shebang. > Acesso em: 9 maio 2019.

KINOSHITA, B. O. **Entendendo as permissões no Linux**. Disponível em: < https://www.vivaolinux.com.br/artigo/Entendendo-as-permissoes-no-Linux.> Acesso em: 9 abr. 2019.

ARRUDA, F. **Entendendo e configurando permissões de arquivos e pastas no Linux**. Disponível em: < https://canaltech.com.br/linux/entendendo-e-configurando-permissoes-de-arquivos-e-pastas-no-linux/.> Acesso em: 9 abr. 2019.

ALECRIM, E. **Entendendo e usando permissões no Linux**. Disponível em: https://www.infowester.com/linuxpermissoes.php. Acesso em: 9 abr. 2019.

PINTO, Pedro. **LINUX – Permissões em ficheiros e directórios**. Disponível em: < https://pplware.sapo.pt/linux/linux-permissoes-em-ficheiros-e-directorios/.> Acesso em: 9 abr. 2019.

SCHIMITZ, D. **Tudo que você queria saber sobre Git e GitHub, mas tinha vergonha de perguntar**. Disponível em: < https://tableless.com.br/tudo-que-voce-queria-saber-sobre-git-e-github-mas-tinha-vergonha-de-perguntar/.> Acesso em: 9 abr. 2019.

GITHUB. **Write better code**. Disponível em: < https://github.com/features/code-review/.> Acesso em: 10 maio 2019.

PYTHONBRASIL. **Sobre Python.** Disponível em: < https://wiki.python.org.br/PerguntasFrequentes/SobrePython. > Acesso em: 10 maio 2019.

BORGES, L. E. . **Python para Desenvolvedores 2ed**. Rio de Janeiro: Edição do Autor, 2010.

BOOTSTRAP. **Bootstrap Expo**. Disponível em: < https://expo.getbootstrap.com/.> Acesso em: 1 maio 2019.

BOOTSNIP. **Bootsnipp - Design elements, playground and code snippets for Bootstrap HTML/CSS/JS framework**. Disponível em:< https://bootsnipp.com/.> Acesso em: 1 maio 2019.

RASPBERRY PI FOUNDATION. **GPIO**. Disponível em: < https://www.raspberrypi.org/documentation/usage/gpio/.> Acesso em: 1 maio 2019.

MARIMOTO, C. **Redes - Guia Pratico 2ed**. Porto Alegre: Sul Editores, 2010.

DIPOL. **PoE - IEE 802.3af e IEE 802.3at**. Disponível em: <https://www.dipol.pt/poe\_-\_iee\_802\_3af\_e\_iee\_802\_3at\_bib746.htm.> Acesso em: 22 maio 2019.