UNIVERSIDADE DE SÃO PAULO

ESCOLA DE ENGENHARIA DE SÃO CARLOS

DEPARTAMENTO DE ENGENHARIA ELÉTRICA E DE COMPUTAÇÃO

Implementação de um servidor de arquivos utilizando Linux embarcado e desenvolvimento de um algoritmo de permutação em blocos

Autor: Robério Soares Nunes

Orientador: Prof. Dr. Evandro Luis Linhari Rodrigues

São Carlos

2016

Robério Soares Nunes

Implementação de um servidor de arquivos utilizando Linux embarcado e desenvolvimento de um algoritmo de permutação em blocos

Trabalho de Conclusão de Curso apresentado
à Escola de Engenharia de São Carlos, da
Universidade de São Paulo

Curso de Engenharia Elétrica

ORIENTADOR: Prof. Dr. Evandro Luis Linhari Rodrigues

São Carlos
2016

Nunes, Robério Soares
N971i Implementação de um servidor de arquivos utilizando Linux embarcado e desenvolvimento de um algoritmo de permutação em blocos / Robério Soares Nunes; orientador Evandro Luis Linhari Rodrigues. São Carlos, 2016.

Monografia (Graduação em Engenharia Elétrica com ênfase em Eletrônica) -- Escola de Engenharia de São Carlos da Universidade de São Paulo, 2016.

1. Linux embarcado. 2. Raspberry Pi. 3. servidor LAMP. 4. interface web. 5. criptografia privada. 6. permutação em blocos. I. Título.

FOLHA DE APROVAÇÃO

Nome: Robério Soares Nunes

Título: "Implementação de um servidor de arquivos utilizando Linux embarcado e desenvolvimento de um algoritmo de permutação em blocos"

Trabalho de Conclusão de Curso defendido e aprovado em 22/11/2016,

com NOTA 9,3 (Nove, Três), pela Comissão Julgadora:

Prof. Associado Evandro Luis Linhari Rodrigues - Orientador - SEL/EESC/USP

Mestre Leonardo Mariano Gomes - Doutorando - SEL/EESC/USP

Mestre Alex Antonio Affonso - Doutorando - SEL/EESC/USP

Coordenador da CoC-Engenharia Elétrica - EESC/USP:
Prof. Associado José Carlos de Melo Vieira Júnior

Agradecimentos

Gostaria de agradecer, primeiramente, aos meus pais, que me guiaram nessa trajetória me dando conforto e carinho.

À minha irmã Patrícia, pelas inúmeras vezes que contei com sua ajuda.

A todos os professores dessa instituição, que compartilharam um pouco de seus conhecimentos durante minha formação.

Ao meu orientador Evandro, que me ajudou na condução desse trabalho, além dos demais conselhos e ensinamentos durante minha graduação.

Resumo

Este trabalho de conclusão de curso visa implementar um servidor de arquivos em um sistema embarcado Linux, utilizando uma Raspberry Pi. O servidor de arquivos é acessado pelo usuário através de uma plataforma web, utilizando um sistema LAMP (Linux, Apache, MySQL e PHP). O acesso e controle dos arquivos é feitos através do *website* criado em PHP. Também é desenvolvido e proposto um algoritmo de permutação em blocos, do inglês *permutation box*. O algoritmo usa chaves escolhidas pelo usuário para realizar a permutação. Tal permutação é realizada de maneira dinâmica e é uma etapa fundamental em qualquer algoritmo de criptografia de chave privada. O *webserver* funcionou como planejado, sendo possível armazenar arquivos de maneira prática, confidencial e com um baixo custo. O algoritmo de permutação também funcionou como planejado, ele foi implementado em linguagem de programação c, sendo ágil e tornando prática a permutação de arquivos e posterior armazenamento no *webserver*.

Palavras-chave: Linux embarcado, Raspberry Pi, servidor LAMP, interface web, criptografia privada, permutação em blocos.

Abstract

This final course work intends to implement a file server in a embedded Linux system, using a Raspberry Pi. The file server is accessed through a web platform, using a LAMP (Linux, Apache, MySQL and PHP). The access and control of the files is made through the website created in PHP. It is also developed and proposed an algorithm of permutation boxes. The algorithm uses keys chosen by the user to do the permutation. Such permutation is done in a dinamic way and is a fundamental step in any private key cryptographic algorithm. The webserver behaved as planned, it was possible to store files in a practical manner, confidential and with a low cost. The permutation algorithm also behaved as planned, it was developed in c programming language, being agile and making practical the process of permutating a file and subsequently storing it on the webserver.

Keywords: Embedded Linus, Raspberry Pi, LAMP server, web interface, private crypto-grafy, permutation boxes.

Lista de Figuras

Lista de Tabelas

Siglas

RPi	*Raspberry Pi*
LAMP	*Linux Apache MySQL PHP*
PC	*Personal Computer*
APP	*Application*
CSS	*Cascading Style Sheets*
ISP	*Internet Service Provider*
DES	*Data Encryption Standard*
AES	*Advanced Encryption Standard*
RSA	*Rivest Shamir Adleman*
P-box	*Permutation Box*
S-box	*Substitution Box*
USB	*Universal Serial Bus*
GPIO	*General Purpose Input/Output*
IoT	*Internet of Things*
GB	*Gyga Byte*
RAM	*Random Access Memory*
LAN	*Local Area Network*
HTTP	*Hypertext Transfer Protocol*

Sumário

Capítulo 1

Introdução

Com o surgimento de servidores de armazenamento de arquivos na nuvem como o DropBox e o Google Drive, acessados através de um PC ou via APP em dispositivos móveis, e a recente popularização de dispositivos embarcados, como o Rasberry Pi, tornou-se acessível a implementação de um sistema caseiro de armazenamento de arquivos de baixo custo, para fins práticos e didáticos. Uma das aplicações de um servidor de arquivos caseiro, rodando em Linux embarcado, é a confidencialidade dos dados armazenados, já que os arquivos não estão em poder de uma empresa. Este trabalho de conclusão de curso visa implementar um sistema de acesso e gerenciamento de arquivos web que possua algum nível de segurança nos dados armazenados.

Com o intuito de garantir uma camada extra de segurança nos arquivos, este trabalho irá desenvolver e propor um algoritmo de permutação personalizado, de maneira que além da camada extra de segurança, sejam feitas análises no campo da criptografia. Serão combinados dois algoritmos de permutação, o primeiro utilizando o Teorema Chinês dos Restos e o segundo será feito através do algoritmo da divisão euclidiana.

O desenvolvimento de um servidor web e de um algoritmo de permutação de arquivos são duas tarefas que, embora distintas, se relacionam diretamente, visto que os arquivos armazenados no servidor poderão ser permutados previamente e despermutados através de sua chave. Por fim, no armazenamento de arquivos é necessário a utilização de algum algoritmo criptográfico para maior segurança dos dados armazenados.

1.1 Motivação

A motivação para este trabalho de conclusão de curso, veio do fato que o preço atual de 1TB de espaço de armazenamento na nuvem custa em torno de USD 9,99/mês. Levando em consideração que o usuário terá que arcar com os custos mensais por tempo indeterminado, uma solução alternativa e com preço similar a uma assinatura anual, é implementar uma solução de armazenamento e acesso remoto caseiros. O custo de um kit completo com um Raspberry Pi custa USD 50,00 e o custo de uma HD externo é atualmente de USD 60,00. A solução caseira tem a vantagem do usuário possuir o equipamento físico (que pode ser usado para outras finalidades em um futuro ou até mesmo revendido).

É possível implementar um sistema de baixo custo em uma Raspberry Pi que conte com *backup* de arquivos em dispositivos de armazenamentos diferentes. Tal backup traria confiabilidade ao sistema e competiria com as soluções já existentes e robustas do armazenamento na nuvem. Além disso, os arquivos armazenados podem ser versados para diferentes sistemas operacionais que utilizam sistemas de arquivos diferentes, como o Linux e o Windowns. Finalmente, a segurança e confidencialidade dos dados armazenados é um fator decisivo na escolha de um sistema de armazenamento, pois a prática de venda de informações de usuários para empresas de propaganda é bastante comum nos dias de hoje.

1.2 Objetivos

Os objetivos desse trabalho de conclusão de curso são destacados a seguir: implementação de um servidor de arquivos web robusto o suficiente para uso contínuo, cujo custo do projeto seja comparável as aplicações comerciais disponíveis. Além disso, a natureza acadêmica desse projeto faz com que um dos objetivos seja o aprendizado de suas ferramentas principais, que tem diversas aplicações, como os sistemas Linux embarcados e as interfaces web. Por último, o objetivo em desenvolver um algoritmo de permutações em blocos que utiliza-se de novos métodos, como o Teorema Chinês dos Restos, é acadêmico, tendo aplicações práticas no armazenamento de arquivos do servidor web.

Capítulo 2

Embasamento Teórico

Este projeto pode ser dividido em duas etapas independentes, mas que se complementam. A primeira etapa é a implementação de um servidor remoto de arquivos, rodando em um dispositivo embarcado com sistema operacional Linux, que é acessado através de uma interface web. Por último, o desenvolvimento e proposta de um algoritmo de permutação, popularmente conhecido como *permutation box*, que é uma das etapas de uma criptografia de chave simétrica. Este capítulo trata brevemente dos principais conceitos envolvidos na implementação de ambas etapas desse projeto.

2.1 Sistemas embarcados

Um sistema embarcado pode ser entendido como um dispositivo eletrônico projetado para ser eficiente em *hardware* e *software* para aplicações específicas (como era utilizado no passado) ou em aplicações mais abrangentes, porém atuante em um escopo bem definido (como os *smartphones* e os sistemas embarcados modernos como o *Beagle Bone Black*). Também é importante notar que os sistemas embarcados estão presentes nas mais diversas aplicações domésticas (televisor, micro-ondas, máquina de lavar) ou industriais (esteiras, robôs, sensores inteligentes).

Cunha em [1], classifica os sistemas embarcados em quatro categorias, a saber:

1. Propósito geral: sistemas embarcados que não tem somente uma aplicação específica, mas pode servir a outras aplicações e é caracterizado pelo alta interação com o usuário.

2. Sistemas de controle: são aplicações em tempo real, que no geral necessitam de

pouca ou nenhuma interação com o usuário. Suas aplicações atendem a sistemas críticos que precisam de alta precisão e fazem uso de diversos sensores para haver redundância. Como exemplo tem-se o freio ABS.

3. Processamento de sinais: são aplicações em tempo real que geralmente fazem a conversão do meio analógio para o meio digital e vice versa. Como exemplo tem-se os modens.

4. Comunicações e redes: sistemas responsáveis por distribuir os sinais dos meios de comunicação corretamente, como a correta separação de duas mensagens que são distribuídas pelo mesmo canal.

Cunha também discute a importância das características de um sistema embarcado no desenvolvimento de projetos em [1]. Tamanho e peso são fatores críticos nesses sistemas, tanto para minimização dos custos, quanto para suas características portáteis. Além disso, o consumo de energia deve ser minimizado, pois muitos desses sistemas trabalham em período contínuo, reduzindo os custos de manutenção. E finalmente, a resistência aos intempéries é um fator chave, pois um sistema embarcado estará sujeito a variações térmicas, de umidade, etc.

2.2 Algoritmos de Criptografia

Branco define criptografia, em [2], como: "Processo de transformação, através de uma chave secreta, de informação legível (mensagem) em informação ilegível (criptograma)". O algoritmo de transformação deve tornar a mensagem (ou *plaintext*) em um criptograma (ou *ciphertext*) que seja resistente a métodos de criptoanálise, ou seja, cujos únicos dois métodos de decriptação conhecidos sejam através do conhecimento da chave de criptografia ou da exaustiva checagem de todas as chaves possíveis (método conhecido como força bruta), que pode levar um número expressivamente grande de anos para se checar todas as chaves. O algoritmo de criptografia pode ser de conhecimento público, pois o segredo que codifica a mensagem deve estar guardado apenas na incerteza da chave. Em computação, a chave é representada por uma sequência de bits.

Os algoritmos de criptografia podem ainda ser classificados como de chave simétrica e assimétrica. O algoritmo de chave simétrica (ou privada) utiliza a mesma chave

pra encriptação e decriptação dos dados, enquanto que o algoritmo de chave assimétrica (ou pública) utiliza chaves diferentes para encriptação de decriptação, como Branco cita em [2].

2.2.1 Criptografia simétrica

Dentre os algoritmos de criptografia simétrica, cifras em blocos (do inglês *block cipher*) é o tipo de algoritmo mais utilizado. Esta classe de algoritmos atua em blocos, encriptando um número definido de dados (geralmente definido em bits). O algoritmo recebe como entrada um bloco de dados da mensagem e retorna um bloco de mesmo tamanho já criptografado [3]. Sendo assim, toda a mensagem é encriptada em blocos de tamanho específico.

Para construir o algoritmo de criptografia em blocos, são necessários vários estágios de transformação dos dados. Em 1949, como mencionado por Stallingsem em [3], Claude Shannon introduziu o conceito de substituição e permutação em cifras em blocos, que são utilizados até hoje em diversos algoritmos, como no AES - Advanced Encryption Standard. As operações que envolvem os blocos de substituição (S-box) e os blocos de permutação (P-box), são explicadas a seguir:

Permutation Box

As P-boxes são nada mais do que uma permutação dos elementos da mensagem de entrada, não havendo alteração no conteúdo dos elementos da mensagem. Como mencionado em [3], os efeitos de "difusão"e "confusão", também introduzidos por Claude Shannon, são desejáveis em uma cifra em blocos. Uma P-box é responsável pelo efeito de difusão, ou seja, cada elemento da mensagem é difundido, espalhado pela sua respectiva cifra em bloco. Vale lembrar que como toda permutação, as P-box são inversíveis, para decriptação da mensagem.

Substitution Box

As S-boxes são responsáveis pelo efeito de confusão na cifra em blocos, pois substituem os elementos da mensagem por novos elementos de mesma categoria, de maneira que a relação entre a mensagem e o criptograma não seja facilmente identificado [4]. A S-box é representada por uma função f, que é bijetora. Sendo A o conjunto das cifras em bloco (ou

uma parte bem definida da cifra, como um *array*), f é definido como $f : A \rightarrow A$, ou seja, uma S-box é uma função bijetora que relaciona duas cifras em blocos. A natureza dessas funções f não é de fácil implementação, já que elas não devem ser lineares nem uniformemente diferenciáveis e ter grau algébrico maior que 3 para serem resistentes aos métodos atualmente conhecidos de criptoanálise, como mencionado em [4].

Efeito Avalanche

Nos algoritmos de criptografia, como o DES - Data Encryption Standard e o AES, são utilizados várias vezes e alternadamente P-box e S-box, no intuito de provocar os efeitos de confusão e difusão. Além disso, outro efeito desejado que também é consequência das vários utilizações de P-box e S-box, é o efeito avalanche. [3] define o efeito avalanche como: "Uma mudança em um bit do dado de entrada ou de um bit na chave provoca a mudança de cerca de metade dos bits da saída", ou seja, a probabilidade de um bit do criptograma ser alterado quando se altera um bit da mensagem ou da chave é de 50%.

2.2.2 Criptografia AES

O *Advanced Encryption Standard* é um algoritmo de chave privada que atua em blocos de 128 *bits* e tem chaves de 128, 192 e 256 *bits*. AES utiliza o algoritmo Rijndael, que foi vencedor do concurso promovido pelo *National Institute of Standards and Technology*, para ser o sucessor do algoritmo DES [5].

AES é utilizado até hoje pelo governo norte americano para criptografia de arquivos sigilosos classificados como "*unclassified*"[5]. Ele é implementando em *software*, *hardware* e *firmware*. Uma aplicação bem difundida do AES é na criptografia dos dados em uma rede *WiFi*.

Conforme explicado em [5], o AES é composto de 4 operações fundamentais em cascata que são iterativamente repetidas entre 10 e 14 vezes. São elas:

1. *SubBytes() Transformation*: uma S-box com boas propriedades de não linearidade. Essa etapa é responsável pelo efeito de confusão.
2. *ShiftRows() Transformation*: uma P-box que desloca horizontalmente os ele-

mentos das linhas do bloco de cifras 4x4. A primeira linha é deixada intacta, a segunda linha é deslocada uma unidade para esquerda, a terceira linha é deslocada duas unidades para esquerda enquanto que a quarta linha é deslocada 3 unidades para esquerda. Essa etapa é responsável pelo efeito de difusão.

3. *MixColumns() Transformation*: essa etapa mixa os elementos das colunas da cifra, os substituindo por novos elementos de maneira biunívoca. Tal etapa é aplicada em todas as colunas da cifra 4x4. Essa etapa é responsável pelo efeito avalanche, pois ao alterar um bit de um elemento de uma coluna, os outros três elementos também são alterados.

4. *AddRoundKey() Transformation*: essa etapa consiste de uma adição booleana,ou seja, uma operação XOR da cifra com uma chave de tamanho 4x4. Essa etapa, em conjunto com a etapa 1. gera o efeito de confusão.

2.3 Teorema Chinês dos Restos

O Teorema Chinês dos Restos é um resultado muito conhecido em Teoria dos Números elementar e tem diversas aplicações práticas ou teóricas. Pelo fato do teorema fazer uso de aritmética modular, é natural que o teorema tenha aplicações em criptografia. Por exemplo, é demonstrado matematicamente em [6] como o uso do teorema pode tornar o algoritmo de decriptação de chave pública RSA (que recebe os sobrenomes de seus criadores Rivest, Shamir e Adleman) até quatro vezes mais eficiente. Isso é possível pois teorema consegue dividir o algoritmo de decriptação RSA em casos menores com custo computacional inferior ao algoritmo de decriptação RSA padrão. Segue o enunciado do Teorema Chinês dos Restos, presente em [7].

Considere n inteiros positivos m_1, m_2,...,m_n dois a dois primos entre si, ou seja, $mdc(m_i, m_j) = 1$, $i \neq j$. Então o sistema de congruências

$$N \equiv a_1 \ mod(m_1)$$
$$N \equiv a_2 \ mod(m_2)$$
$$...$$
$$N \equiv a_n \ mod(m_n)$$

admite solução N, quaisquer que sejam os inteiros a_1, a_2,...,a_n. Além disso N é único módulo $M = m_1.m_2...m_n$ (se $N \equiv b_1 \ mod(M)$ e $N \equiv b_2 \ mod(M)$, então $b_1 \equiv b_2 \ mod(M)$).

2.4 Algoritmos de permutação

Nesse trabalho de conclusão de curso, é proposto um novo algoritmo para geração de uma família de P-boxes utilizando-se da combinação de dois algoritmos de permutação, o primeiro sendo baseado no Teorema Chinês dos Restos e o segundo sendo um algoritmo que relaciona biunivocamente um número $0 \leq x < n!$ a uma permutação dos números $0, 1, ..., n-1$, ou seja, x é a chave que decripta uma permutação dos índices de um vetor de n elementos.

2.4.1 Permutação baseada no Teorema Chinês dos Restos

Deseja-se criar uma *permutation box* em um bloco de bytes de um arquivo, cujos *bytes* são permutações dos *bytes* do arquivo original. Dito isso, a estratégia principal em criar essa P-box usando o teorema, baseia-se no fato que N é único módulo M, ou seja, para cada n-upla (a_1,..,a_n), com $0 \leq a_i < m_i$, existe um único N com $0 \leq N < M$. Portanto, um arquivo, que pode ser entendido como um vetor unidimensional, pode ser permutado de acordo com a regra do Teorema Chinês dos Restos, sem sobreposições, pois a função de permutação é uma função bijetora.

No algoritmo da P-box utiliza-se um sistema de duas congruências tais que $m_1.m_2$ seja maior ou igual ao tamanho da P-box em *bytes*. A partir de agora, a P-box será visualizada de maneira retangular (apesar dela continuar sendo um vetor unidimensional para o computador), como um retângulo de lados m_1 e m_2, ou uma matriz de m_1 colunas e m_2 linhas. O *byte* de índice i da cifra em bloco inicial, gera as duas coordenadas a_1 e a_2 da cifra em bloco do criptograma, tal que $i \equiv a_1 \ mod(m_1)$ e $i \equiv a_2 \ mod(m_2)$, portanto o *byte* de índice i será movido para a coluna a_1 e linha a_2 no arquivo criptografado, isso é o equivalente a dizer que o *byte* será movido para posição $a_1 + m_1.a_2$ no arquivo criptografado (note que para existir alinhamento com os índices do computador, a coluna 1 da matriz de visualização do arquivo é chamada de coluna 0, e a linha 1 é chamada de linha 0).

O Teorema Chinês dos Restos garante que essa permutação dos *bytes* seja feita

de maneira única, havendo a possibilidade de decriptação sem perda de informações. além disso, o tamanho original do aquivo se mantém. O exemplo a seguir ilustra como um arquivo de 27 *bytes* pode ser criptografado usando-se $m_1 = 5$ e $m_2 = 6$:

Arquivo inicial na forma vetorial:

$a_1\ a_2\ a_3\ a_4\ a_5\ b_1\ b_2\ b_3\ b_4\ b_5\ c_1\ c_2\ c_3\ c_4\ c_5\ d_1\ d_2\ d_3\ d_4\ d_5\ e_1\ e_2\ e_3\ e_4\ e_5\ f_1\ f_2$

Forma matricial:

$$\begin{array}{c} a_1\ a_2\ a_3\ a_4\ a_5 \\ b_1\ b_2\ b_3\ b_4\ b_5 \\ c_1\ c_2\ c_3\ c_4\ c_5 \\ d_1\ d_2\ d_3\ d_4\ d_5 \\ e_1\ e_2\ e_3\ e_4\ e_5 \\ f_1\ f_2 \end{array}$$

Arquivo Criptografado na forma vetorial:

$a_1\ b_2\ c_3\ d_4\ e_5\ f_1\ a_2\ b_3\ c_4\ d_5\ e_1\ f_2\ a_3\ b_4\ c_5\ d_1\ e_2\ d_3\ a_4\ b_5\ c_1\ d_2\ e_3\ e_4\ a_5\ b_1\ c_2$

Forma matricial:

$$\begin{array}{c} a_1\ b_2\ c_3\ d_4\ e_5 \\ f_1\ a_2\ b_3\ c_4\ d_5 \\ e_1\ f_2\ a_3\ b_4\ c_5 \\ d_1\ e_2\ d_3\ a_4\ b_5 \\ c_1\ d_2\ e_3\ e_4\ a_5 \\ b_1\ c_2 \end{array}$$

Nesse trabalho, o algoritmo de permutação usando o Teorema Chinês dos Restos sempre usará dois números consecutivos n e $n+1$ para expressar o arquivo em fora matricial, pois $mdc(n, n+1) = 1$ para todo n inteiro.

Nota-se que uma limitação desse algoritmo é o fato dos *bytes* continuarem na mesma coluna, comprometendo a complexidade da permutação dos *bytes*, além da permutação conter diversas linearidades, pela própria natureza modular do Teorema Chinês dos

Restos. Outra característica da permutação, é que os elementos de uma linha tendem a ser distribuídos nas outras linhas, pois o algoritmo move os elementos das linhas para alguma diagonal,na representação matricial. Finalmente, é importante destacar que esse algoritmo de permutação utilizando o teorema chinês dos restos pode ser utilizado em blocos de *bytes* de qualquer tamanho.

2.4.2 Permutação de um vetor de n elementos

Para permutar os índices de um vetor de n elementos, basta associar um número $0 \leq x < n!$ a uma permutação de $0,1,...,n-1$ de maneira biunívoca. Tal permutação é feita conforme o seguinte pseudo código:

Para $i = n-1; i > -1; i$

$nova\_posicao[i] = quociente\ de\ x/i!$

$x = resto\ de\ x/i!$

No pseudo código acima, $nova\_posicao[i]$ guarda a posição que i deve ocupar na permutação de um vetor de i elementos. Para construção da permutação, inicia-se com um vetor de saída vazio, coloca-se o elemento de índice i na posição referente a $nova\_posicao[i]$, em seguida, considera-se apenas as $i-1$ posições vazias restantes e repete-se o processo iterativamente.

Esse algoritmo de permutação é uma bijeção pois ele associa um número $0 \leq x < n!$ aos quocientes $q[i] = x/i!$, para $0 \leq i < n$. Ora, pelo algoritmo da divisão de Euclides, sabe-se que:

$$x = q[i].i! + resto\ da\ divisao$$

Como o resto da divisão é definido iterativamente, pode-se escrever a expressão:

$$x = q[i].i! + q[i-1].(i-1)! + ... + q[1].1! + q[0].0!$$

Como o quociente e resto na divisão de Euclides são únicos, conclui-se que cada inteiro x é biunivocamente associado aos quocientes $q[i],...,q[1],q[0]$.

Capítulo 3

Materiais e Métodos

3.1 Materiais

3.1.1 Raspberry Pi

O Raspberry Pi foi lançado em fevereiro de 2012 pela *Raspberry Pi Fundation*, situada no Reino Unido. O intuito do projeto era criar um computador de baixo custo e alta performance para ser usado de maneira didática, criativa e recreacional. O computador foi desenvolvido pensando-se no conteúdo digital que poderia ser criado pelos seus usuários de maneira simples e prática, incentivando o desenvolvimento de produtos computacionais [8].

Figura 3.1: Raspberry Pi 3 em um case de plástico

Atualmente a versão mais recente do computador é o Raspberry Pi 3 Modelo B, como

visto na figura 3.1, lançado em fevereiro de 2016. Entre as novidades dessa versão, estão as inclusões de *wireless onboard* 802.11n e Bluetooth 4.1. O Custo de um Raspberry Pi 3 é de USD 35,00. Suas principais características são as seguintes, conforme mencionado em [8].

- Processador de 1.2GHz 64-bit quad-core ARMv8 CPU (900Mhz presentes Rpi 2);
- 802.11n Wireless LAN;
- Bluetooth 4.1;
- Bluetooth Baixa Energia (BLE);
- 1GB RAM;
- 4 portas USB;
- 40 pinos GPIO;
- Porta HDMI;
- Porta Ethernet;
- Slot para cartão Micro SD;
- Núcleo gráfico VideoCore IV 3D;

Nota-se que o Raspberry Pi é um sistema embarcado com um poder de processamento capaz de atender diversos projetos. Sua conectividade através de *WiFi, bluetooth, ethernet cable*, 4 portas USB e 40 pinos GPIO (portas de entrada e saída) o tornam pronto para desenvolver vários projetos em eletrônica e voltados à IoT. Finalmente, seu núcleo gráfico VideoCore IV 3D e sua porta HDMI o torna capaz de criar interfaces com o usuário de boa qualidade.

Fonte de alimentação

É recomendado uma fonte de alimentação de 5V 2,0A para alimentar o Raspberry Pi 3, nesse projeto foi utilizado uma fonte da marca AOC com 100-240V de tensão de entrada a 50/60Hz e corrente de entrada de até 0,4A, como visto na figura 3.2. O conector de saída é micro-USB com 5V de tensão de saída e corrente média máxima suportada de 2,2A.

Figura 3.2: Fonte chaveada micro-USD com 5V@2,2A de saída

Cabo de rede e *pen drive*

Para estabelecer a conexão entre o RPi 3 e o computador utilizado para desenvolvimento deste projeto de formatura, foi utilizado um cabo de rede de 1 metro de comprimento, como na figura 3.3. Além disso, foi utilizado um *pen drive* USB 2.0 de 4GB da marca SanDisk como dispositivo de armazenamento de arquivos.

Figura 3.3: Cabo de rede e dispositivo de armazenamento

3.2 Métodos

O diagrama de blocos da figura 3.4 resume as conexões estabelecidas entre os componentes do servidor de arquivos web. Nota-se que o Linux embarcado, que roda no Raspberry Pi, faz todo o processamento e gerenciamento de acesso do servidor web, enquanto que o computador do usuário faz a encriptação dos arquivos a serem armazenados no dispositivo móvel e a decriptação dos arquivos baixados do servidor. Em seguida, o funcionamento de cada componente é explicado para melhor entendimento do projeto.

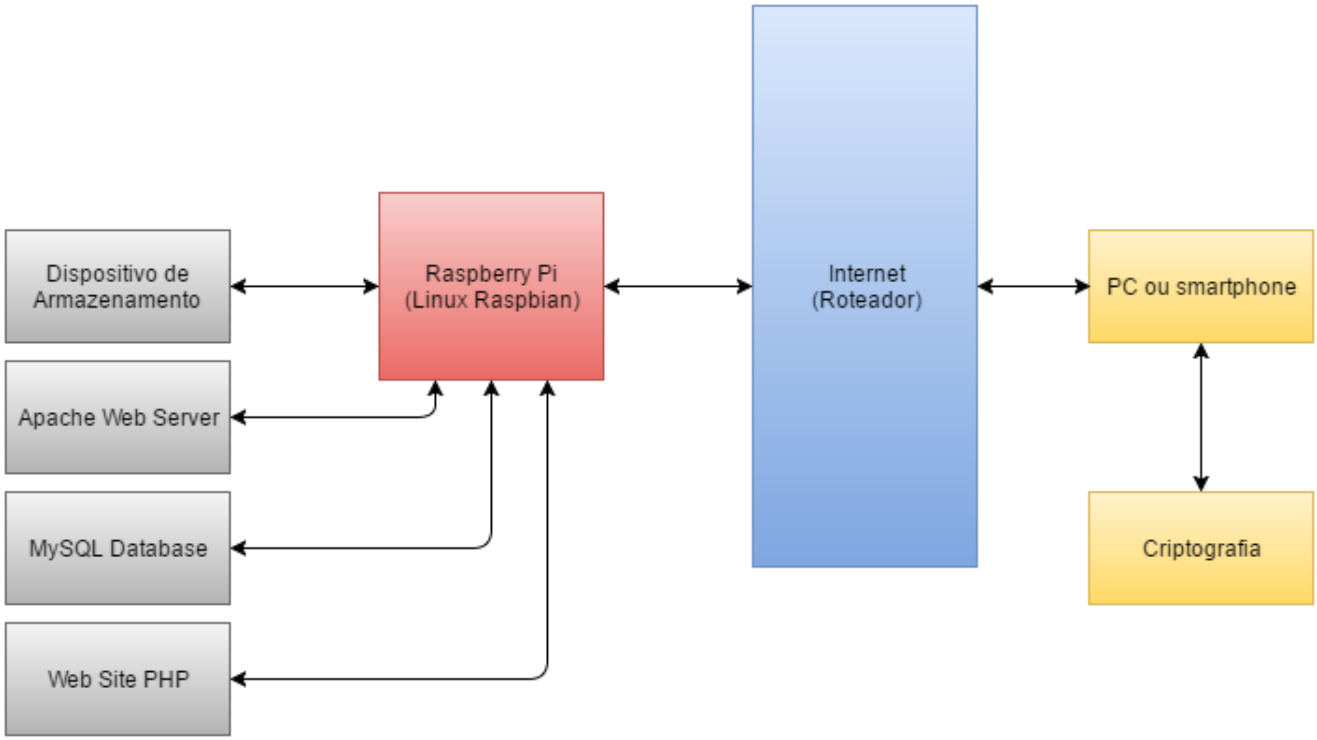

Figura 3.4: Diagrama de blocos do sistema

3.2.1 Configurando o Raspberry Pi

Após a instalação do Raspbian Jessie (versão do Linux Rasbian para RPi), foram necessárias algumas configurações no sistema operacional, como seguem:

- Atualização do gerenciamento de pacotes: através do comando "*sudo apt-get update*";
- Ampliar o uso de memória do cartão SD: disponível no menu acessível através do comando 'sudo raspi-config';
- O usuário administrador padrão do Raspbian é o 'pi' cuja senha é 'raspberry'. Como

o RPi irá ser utilizado como um servidor web, deve-se alterar esta senha através do comando *"sudo passwd"* para evitar que possíveis invasores não tenham acesso total ao dispositivo.

Para armazenar os arquivos que serão upados no servidor remoto, é necessário configurar o dispositivo de armazenamento USB (flash drive ou HD externo). Deve-se formatar o dispositivo e usar os sistemas de arquivos nativo do linux para evitar problemas de compatibilidade, como o Ext4 ou Ext3. O formato Fat32, apesar de ser suportado pelo linux, não é inteiramente compatível, gerando conflitos na escrita de arquivos em suas pastas [10].

O dispositivo USB é externo ao sistema operacional, para que o servidor Apache tenha direitos de escrita nas pastas do dispositivo, é necessário que as pastas do dispositivo sejam do mesmo usuário que o Apache. Como o usuário do Apache é o www-data, deve-se usar o comando *chown* na pasta desejada: '*sudo chown www-data -R /pastadesejada*'. Logo após, é necessário dar direitos de escrita, leitura e execução aos donos, grupos e outros usuários das pastas do dispositivo removível, através do comando *chmod*: '*sudo chmod -R 777 /pasta desejada*'. O número 7 que é $(111)_2$ indica que está sendo atribuído os 3 direitos, e finalmente, cada número 7 é o direito do dono, do grupo do dono e de outros usuários, respectivamente.

O Raspberry Pi não tem um recurso nativo de montagem de dispositivos móveis automaticamente (o drive só é montado automaticamente no momento que o dispositivo é introduzido na porta USB). Para montagem automática do disposivo [11], é necessário alterar o arquivo *fstab* na pasta '*/etc/*', incluindo a linha:

```
/dev/sda1 /mnt/usb ext4 defaults,nofail 0 0
```

Sda1 indica o nome do primeiro dispositivo removível montado, /mnt/usb é a pasta de montagem, Ext4 é o sistema de arquivos do dispositivo, *defaults* seta as configurações de usuário, leitura, escrita, etc como padrões, nofail deve ser utilizado, pois caso o dispositivo de armazenamento esteja removido no momento do *boot* do Raspbian, o *boot* não será interrompido, caso contrário o mesmo será interrompido, causando grandes transtornos, especialmente aos usuários que não utilizam monitor dedicado ao RPi. Os números 0 são outras configurações *default* que não interferem diretamente no funcionamento do servidor web.

3.2.2 Servidor Apache 2

Apache é o servidor HTTP mais utilizado no mundo. Ele foi lançado no ano de 1995 e em 1996 já era líder de mercado [12]. Apache é um projeto de código aberto que é desenvolvido por empresas e colaboradores independentes por todo o mundo. Apache é responsável pelo processamento dos pedidos de acesso ao servidor web, retornando o conteúdo web solicitado. Geralmente o tráfego de dados é feito através da porta 80 (presente no roteador que hospeda o servidor), ou porta 8080.

Para instalar o Apache 2.4, sua versão mais recente, no raspberry pi, basta digitar o seguinte comando na linha de comando:

```
sudo apt-get install apache2 -y
```

Após a instalação ser concluída, é necessário verificar o funcionamento do *web server*, através do endereço http://localhost/ no *browser* do RPi ou através do arquivo index.html, que é criado automaticamente na pasta /var/www/html/. Caso a instalação seja concluída com sucesso, a imagem 3.5 aparecerá no *browser*, como descrito em [13].

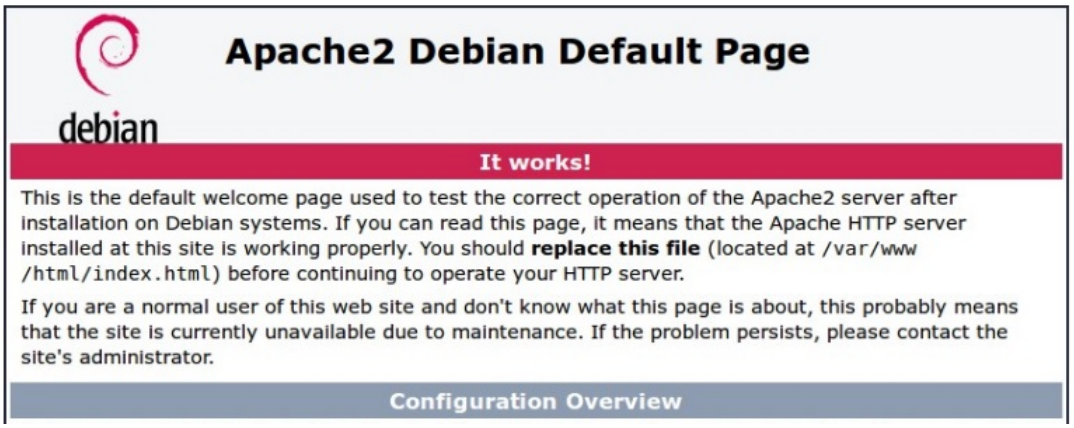

Figura 3.5: Página de testes do Apache

3.2.3 PHP 5

Segundo [13], PHP é responsável por pré-processar conteúdo web dinâmico, que é mostrado no *browser* quando requisitado. Conteúdo dinâmico é essencial para construção da interface gráfica do servidor de arquivos. Além disso, PHP é compatível com Apache, tornando-o uma opção viável para esse projeto de conclusão de curso. Para instalar o PHP 5 e suas bibliotecas compatíveis com Apache 2, utiliza-se o seguinte código:

sudo apt-get install php5 libapache2-mod-php5 -y

Após instalado, deve-se alterar a página inicial do servidor web, substituindo o arquivo *index.html*, na pasta /var/www/html/, pelo arquivo *index.php*. Finalmente, para configuração dos parâmetros dinâmicos do php, deve-se editar o arquivo 'php.ini', presente em /etc/php5/apache2/. Os parâmetros 'upload_max_filesize' e 'post_max_size' foram setados em 1GB, com isso, limita-se o tamanho máximo de upload de um arquivo no servidor em 1GB. A intenção com a imposição desse limite é evitar o *upload* de arquivos durante longos períodos.

3.2.4 MySQL 5 e phpMyAdmin

Para armazenamento dos usuários e senhas que concedem acesso ao servidor de arquivos, foi utilizado o banco de dados MySQL, da Oracle. MySQL é um banco de dados de código aberto mais popular do mundo, segundo o site oficial da Oracle [14]. A instalação do MySQL no Linux é feita através do seguinte comando executado no terminal, no momento da instalação é definido uma senha de acesso ao MySQL.

sudo apt-get install mysql-server php5-mysql -y

Para gerenciamento dos bancos de dados criados no MySQL, é necessário uma interface com o usuário. O *software* utilizado para criação e gerenciamento da base de dados foi o phpMyAdmin. O phpMyAdmin é um programa escrito em PHP que roda localmente, porem é acessado através de um *browser*. A instalação do phpMyAdmin é feita através da execução do seguinte código:

sudo apt-get install phpmyadmin

No momento da instalação, é necessário que o phpMyAdmin seja linkado ao MySQL, para isso, é necessário saber o usuário e senha do MySQL, o usuário padrão é 'root' e a senha é definida no momento da instalação do banco de dados. A figura 3.6 mostra a interface gráfica do phpMyAdmin, no menu a esquerda está selecionada "Tabela", a tabela do MySQL que armazena os usuários e senhas. O conteúdo de "Tabela"pode ser visto na parte central da figura. A tabela contém um ID para buscas eficientes, o usuário, a senha e o e-mail do usuário.

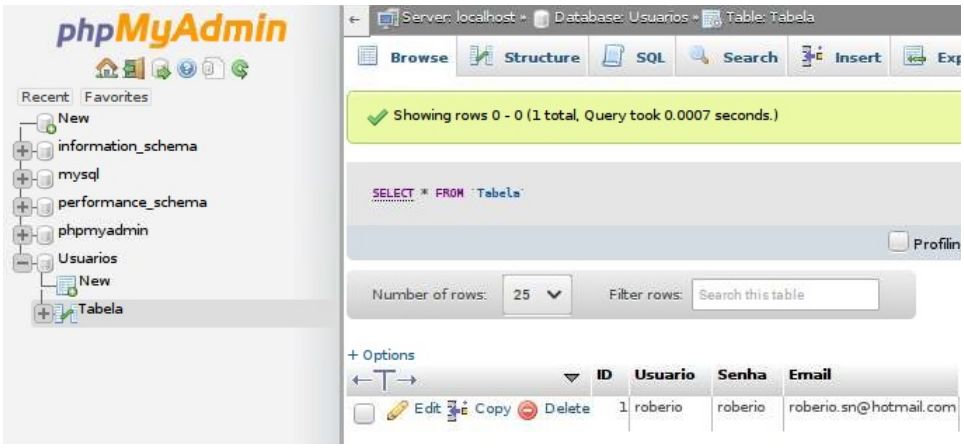

Figura 3.6: Interface do phpmyadmin.

3.2.5 Construindo o Site

A figura 3.7 contém o diagrama de blocos do servidor de arquivos web, para melhor entendimento do funcionamento de como o gerenciamento de arquivos na base é feito.

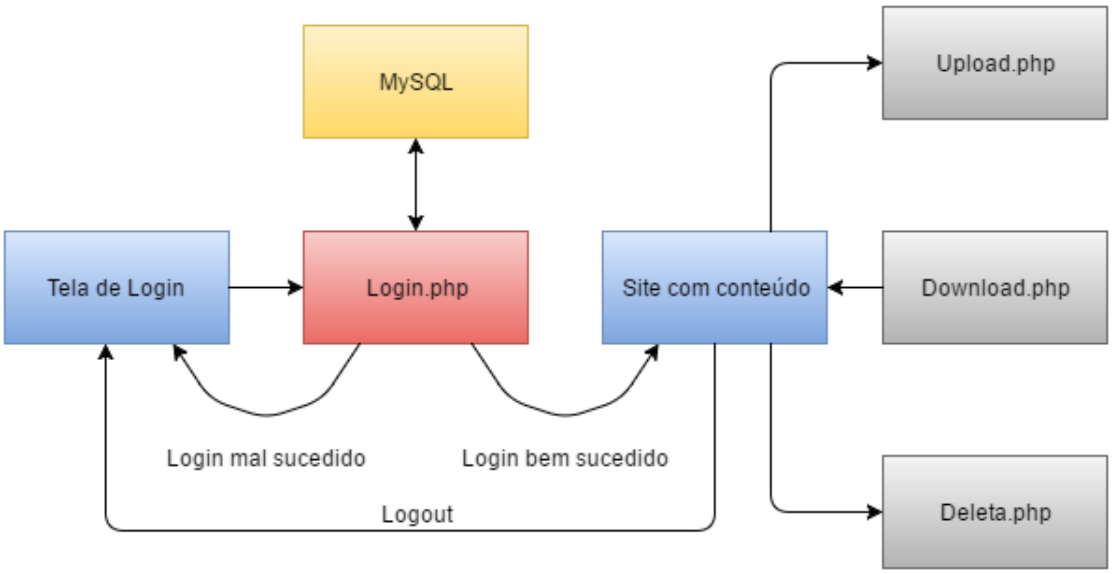

Figura 3.7: Diagrama de blocos do servidor web.

Tela inicial e login

A página inicial do site é, por convenção, http://hostname/index.php, ou seja, a 'Tela de Login', como mostrado na figura 3.8. O usuário deve inserir login e senha previamente cadastrados no banco de dados MySQL para entrar no site, os dados são então enviados para o *script* 'login.php', onde é verificado junto a base de dados se o usuário e senha são válidos.

A figura 3.2 mostra a tela de login.

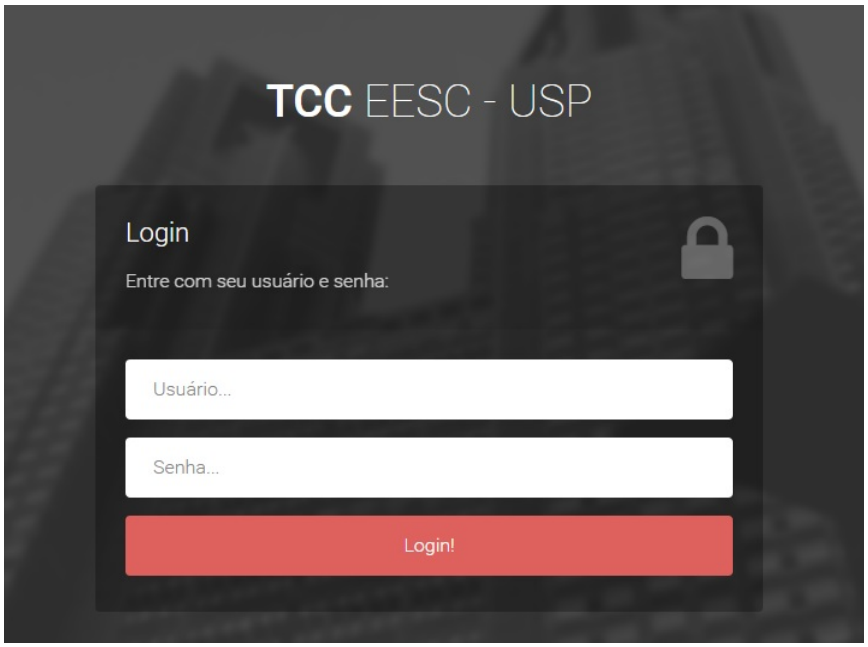

Figura 3.8: Página de Login.

A página 'login.php' recebe os dados e consulta a base de dados no MySQL, caso os dados de login estejam corretos, o usuário é direcionado para o site com conteúdo ('site.php'), caso contrário, ele é redirecionado para tela de login novamente.

Para criação dos *scripts* 'index.php', 'login.php' e 'logout.php' foi utilizado [15] como algoritmo de referência. Além disso, os aplicativos Learn HTML e Learn PHP ([16] e [17]) foram úteis para aprendizado e familiarização com a linguagem.

Bootstrap 3

Bootstrap é o mais popular *web framework*, ou seja, uma biblioteca contendo diversos objetos que melhoram a aparência e navegabilidade em *web sites*. Ao utilizar um *framework*, o site fica com uma aparência profissional, pelo fato dos *frameworks* terem sido desenvolvidos por *web designers*. As vantagens são diversas, dentre elas: ser grátis, de rápida implementação, vasta disponibilidade de *templates* que atendem a diversos projetos e os diversos objetos como botões, tabelas e fontes, que estão embutidas no Bootstrap. O Bootstrap pode ser instalado, mencionado em [18] ou pode ser utilizado *templates* específicos. Nesse

trabalho de conclusão de curso, foi utilizado o *template* presente em [19] na página inicial e demais páginas do web site. Por último, é importante destacar a característica 'responsiva' do Bootstrap, ou seja, seus elementos (tabelas, imagens, blocos de textos, etc) se adaptam ao tamanho e formato da tela, sendo de extrema utilidade nos diversos dispositivos móveis como *tablets* e *smartphones*.

Funcionalidades do web site

Uma vez logado no site, o usuário pode escolher entre upar mais arquivos, realizar downloads e deletar arquivos existentes. Quando é realizado upload de arquivos (na pasta /mnt/usb/E010/UPLOAD/), é gerado um arquivo de backup (na pasta /mnt/usb/E010/BKG/), para proteção contra arquivos corrompidos. Além disso, o usuário pode realizar logout, clicando no botão "Sair"direito superior. A Figura 3.9 mostra a interface do site, incluindo a tabela com os arquivos existentes no servidor. Nota-se que a Tela de Login e a página principal do site tem o mesmo *template*, que foi criado através de [19].

Figura 3.9: Interface do web server.

Na imagem 3.9, há uma tabela com o nome, tamanho e a opção de deletar cada arquivo do servidor. Os nomes dos arquivos contém *links* para *download* dos mesmos, cujo método está presente em [20], enquanto que "Deletar"contém o link do *script* que deleta seu respectivo arquivo. No meio da tela, existe um *form* que faz *upload* de arquivos, este *form* foi

baseado no algoritmo presente em [21]. Por último, após realizar um *login*, uma mensagem de "Bem Vindo"é mostrada no topo da tela, que some caso a página seja atualizada. Após realizar o *upload* e deletar algum arquivo, mensagens também são mostradas confirmando as ações.

Uma função que merece destaque no desenvolvimento do website é a função *session()* em PHP. Ela cria uma sessão diferente para cada login executado com êxito, caso o usuário fique um tempo inativo, a sessão é encerrada e é necessário realizar o login novamente. Ao clicar no botão *sair*, no canto superior direito, a sessão também é encerrada.

A função *session()* é especialmente útil para barrar invasores que tentem acessar o site sem fazer login bem sucedido, para isso no começo de cada *script*, é feito um *check* para detectar se alguma sessão foi iniciada, caso contrário, a página é imediatamente redirecionada para a Tela de Login. Maiores detalhes do funcionamento do *webserver*, estão presentes no Apêndice, com os códigos comentados.

3.2.6 Construindo a *Permutation Box*

Utilizando os dois algoritmos presentes na seção 2.4, foi criado um algoritmo que gera uma família de P-boxes, cada P-box é gerada usando-se 4 chaves de tamanho $20! = 2432902008176640000 \simeq 2^{62}$, ou uma chave de tamanho $20!^4 \simeq 2^{245}$.

O algoritmo atua em um bloco de 400 *bytes*, ou, uma matriz de 20x20. As quatro chaves de tamanho 20!, são utilizadas para gerar 4 permutações de vetores com 20 elementos. 2 vetores de 20 elementos permutados são utilizados para permutar as linhas da matriz 20x20 e os outros 2 vetores permutados são utilizados para permutar as colunas da matriz 20x20. A permutação baseada no Teorema Chinês dos Restos é utilizada para permutar os 400 elementos da matriz. O algoritmo da P-box é descrito a seguir:

1. Permuta-se o bloco inicial de 20x20 com o Teorema Chinês dos Restos.
2. Permuta-se as linhas e colunas do bloco resultante do *item 1* usando-se 2 chaves.
3. Transpõe-se o bloco resultante do *item 2*.
4. Permuta-se o bloco resultante do *item 3*. com o Teorema Chinês dos Restos.
5. Permuta-se as linhas e colunas do bloco resultante do *item 4*. usando-se 2 chaves

restantes.

Como mencionado em 2.4, o passo 1. distribui os elementos de uma linha qualquer da matriz por todas as 20 linhas, mantendo os elementos de cada coluna preservados na mesma coluna. O passo 2. apenas permuta os elementos de cada linha e cada coluna, sem mixar linhas e colunas diferentes. O resultante de 2. é a preservação dos elementos em suas respectivas colunas (mesmo que a posição coluna tenha sido permutada) porém os elementos de cada linha foram espalhados pelas demais linhas, portanto 2. não preserva os elementos em suas respectivas linhas.

3. transpõe a matriz, ou seja, troca linhas por colunas, em seguida 4. e 5. são repetições de 1. e 2. (com chaves possivelmente diferentes). O resultado de 5. é, portanto, a não preservação dos elementos de cada linha e cada coluna. O conteúdo das linhas e colunas são portanto todas mixadas de maneira a não preservar seus estados iniciais.

Existem, portanto, $20!^4 \simeq 2^{245}$ P-box pela definição acima. A qualidade da P-box depende das escolhas das chaves, podendo haver alta ou baixa correlação entre a matriz de entrada e saída do algoritmo.

O algoritmo descrito acima foi implementado em linguagem de programação c, ele aplica uma P-box em cada bloco de 400 *bytes* de um arquivo, de acordo com a escolha das chaves pelo usuário e salva o arquivo resultante. O algoritmo também realiza processo inverso, de decriptar o arquivo, como mostrado na figura 3.10.

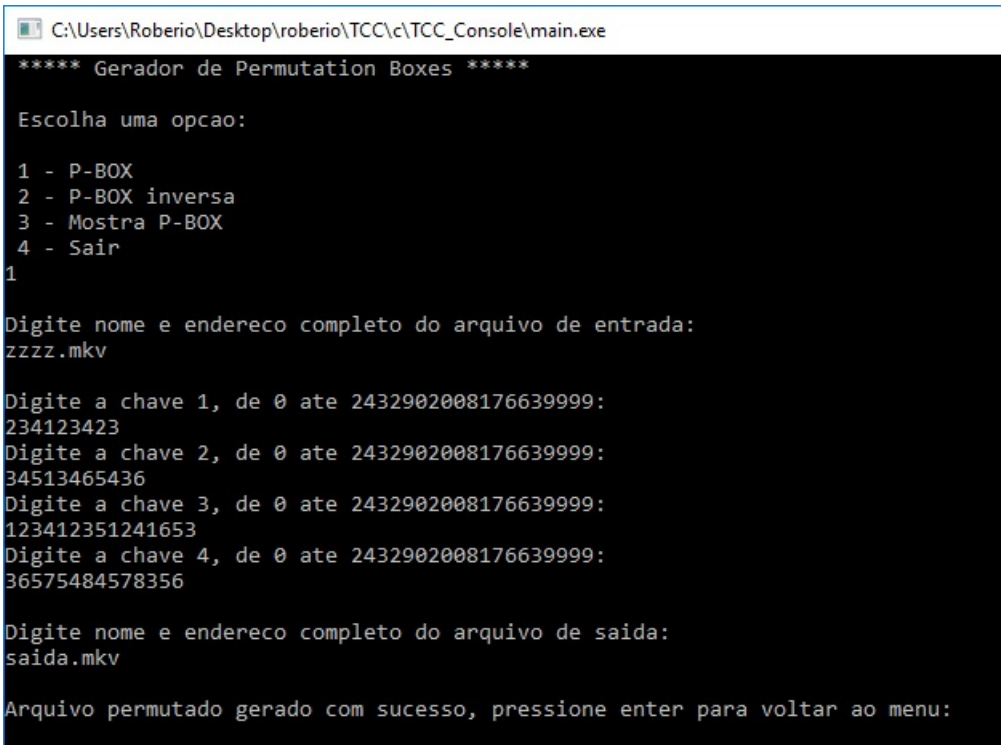

Figura 3.10: Exemplo de criptografia utilizando a P-box

O programa também mostra a P-box utilizada, através da opção ’3’ no menu, como na figura 3.11.

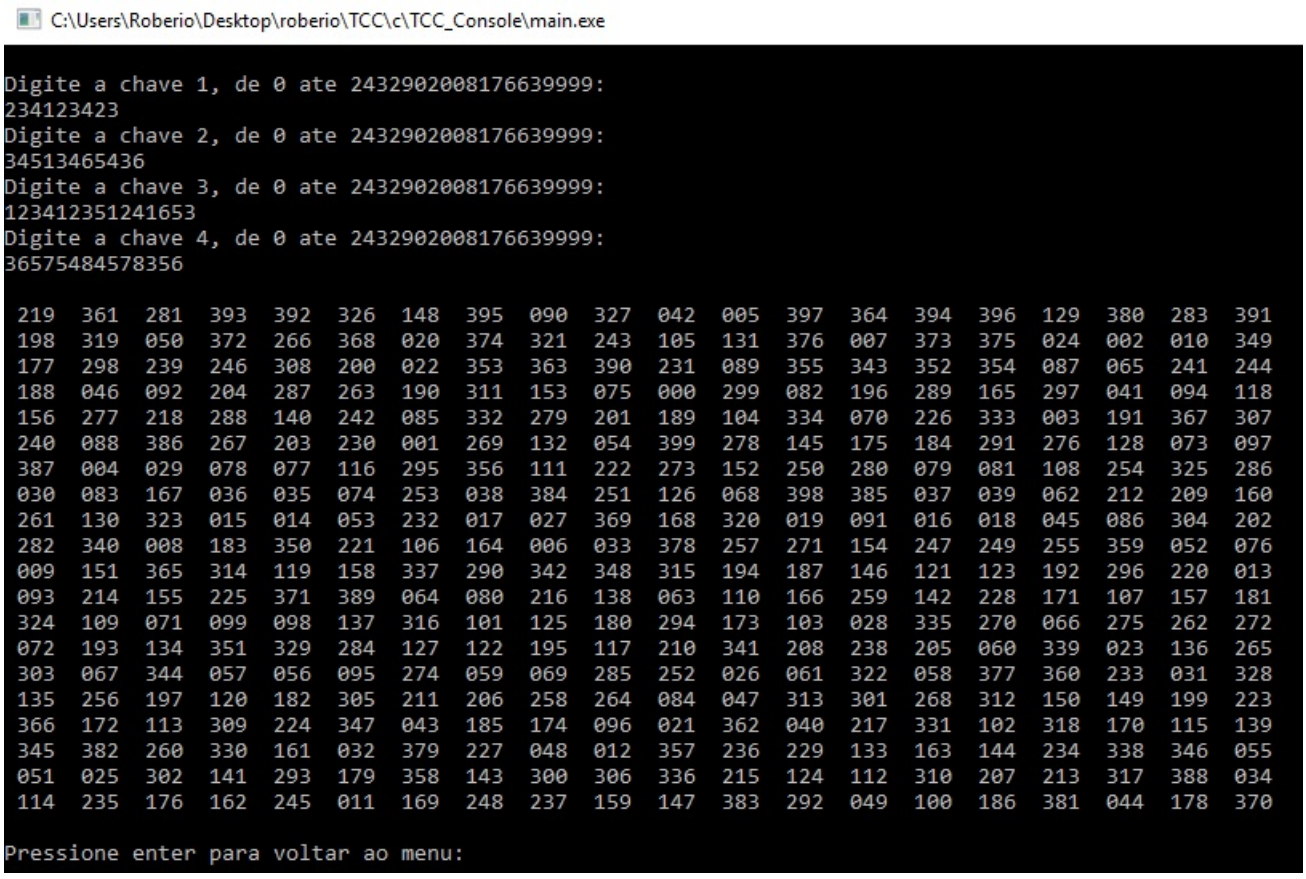

Figura 3.11: Exemplo de P-box

Capítulo 4

Resultados

4.1 Servidor de arquivos

O servidor web funcionou corretamente em diversas plataformas, e navegadores. Foram realizados testes no Google Chrome, Epiphany Web Browser, Microsoft Edge e Safari. O *template* do Bootstrap utilizado mostrou-se compatível com todos esse navegadores citados. Em nenhum navegador houve problemas com o *login*, *upload*, função deletar e *download* de arquivos. O servidor web também foi testado em *smartphones* com sistema operacional Android e iOS, onde pode ser testado a característica responsiva do *template*, como mostra a figura 4.1. Observa-se que a tabela com os nomes dos arquivos também é responsiva.

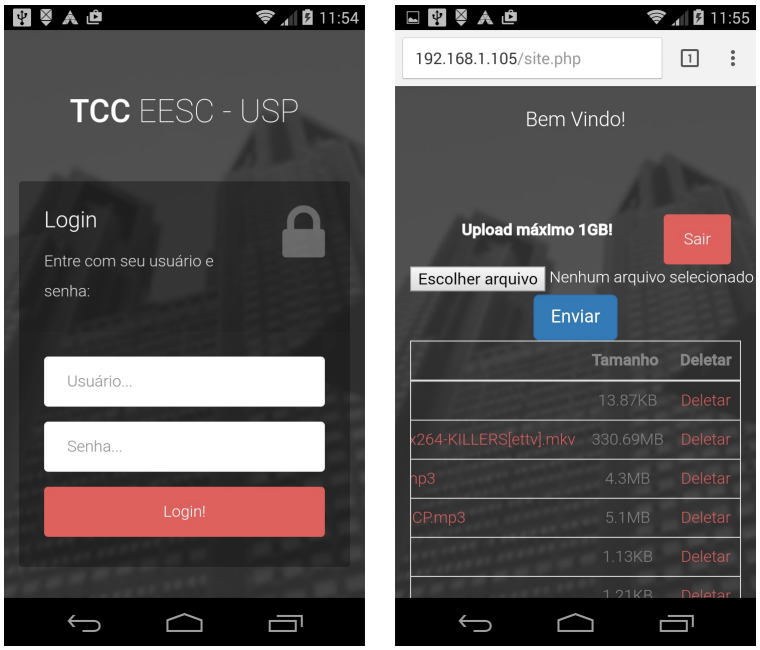

Figura 4.1: Tela de Login e Site com conteúdo visto de um *smartphone* Android

Foram realizados testes de download e upload utilizando uma internet compartilhada de velocidade 10Mbps. Obteve-se taxas de download em torno de 1MB/s, velocidade compatível com o plano de internet contratado.

Não foram observados erros durante a função de deletar arquivos, sendo o processo de deletar arquivos executado rapidamente, mesmo para arquivos de 300MB. Adicionalmente, as pastas contendo os arquivos padrões e a pasta de *backup* estavam sempre sincronizadas, ou seja, quando um arquivo é deletado, seu respectivo *backup* é deletado também, pois a pasta de *backup* não deve acumular arquivos que não são mais desejados.

Mesmo configurando a diretriz *"Upload_max_filesize"* para 1GB, no arquivo 'php.ini', ou seja, ao limitar o tamanho máximo de *upload* de arquivos em 1GB, o servidor web aceitou os pedidos de *upload*. Caso a ferramenta estivesse funcionando como se deseja, o *upload* de arquivos de tamanho superiores a 1GB deveria ser barrado de imediato.

Também não foi observado quedas de conecção no *website*, ou seja, o site sempre estava disponível quando era feito um pedido de acesso. Por último, o *website* consegue barrar o *upload* de arquivos com nomes repetidos de maneira satisfatória, isso ocorre para evitar a sobrescrição de arquivos, uma mensagem de alerta no topo da página alerta a tentativa malsucedida.

4.2 P-box

O algoritmo desenvolvido que gera uma P-box de 20 linhas e 20 colunas foi implementado com êxito em linguagem de programação *c*. Inicialmente o algoritmo foi testado usando-se arquivos de tamanho pequeno, na casa dos poucos mega *bytes*. O teste consistia em gerar um arquivo de mídia com seus blocos de 400 *bytes* permutados, através da escolha aleatória de 4 chaves. Verificava-se que o arquivo de mídia resultante perdia suas características, podendo apenas ser lido como arquivo binário. Em seguida-se realiza-se o processo inverso, "despermutação", recuperando o arquivo inicial.

Foram realizados testes em arquivos de tamanhos maiores. O arquivo de vídeo big.mkv de 712MB foi permutado e medido o tempo necessário para se realizar a permutação, como pode ser visto na figura 4.2. O tempo de execução foi medido através da função *clock()*

pertencente à biblioteca '*time.h*'.

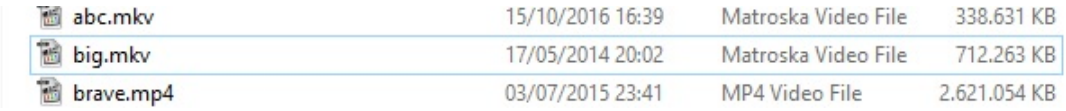

Figura 4.2: Arquivo big.mkv a ser permutado.

O tempo de execução foi medido em 2517ms, ou 2,5 segundos, como mostra a figura 4.3. Portanto a velocidade de permutação pode ser estimada em $712MB/2,5s = 284,8MB/s$. O computador utilizado para realizar a permutação tem um processador Core i7-4500U @ 1.80GHz com 8GB de memória RAM e sistema operacional Windows 10.

```
Digite nome e endereco completo do arquivo de entrada:
big.mkv

Digite a chave 1, de 0 ate 2432902008176639999:
243635657
Digite a chave 2, de 0 ate 2432902008176639999:
45734687248
Digite a chave 3, de 0 ate 2432902008176639999:
2457245725462
Digite a chave 4, de 0 ate 2432902008176639999:
2546245624562

Tempo de execucao: 2517.000000
```

Figura 4.3: Medição do tempo de execução através da função *clock()*

Após implementação bem sucedida do algoritmo em *c*, foram realizados vários testes de performance para diferentes tamanhos de arquivo, resumidos na tabela 4.1. Nota-se que a velocidade de encriptação, medida em MB/s, decresce conforme o tamanho do arquivo encriptado aumenta. Para arquivos pequenos, a velocidade é pouco abaixo dos 350MB/s e para arquivos de algumas centenas de MB, a velocidade é próxima dos 285MB/s. Para arquivos de tamanho maiores que 750MB, o programa de encriptação travou, sendo esse o tamanho máximo de arquivos que puderam ser encriptados.

A qualidade da permutação gerada depende unicamente da escolha das chaves. Chaves que são múltiplas de 19!, por exemplo, são permutações muito fracas em que o único elemento permutado é o elemento da posição 20 do vetor inicial.

A etapa *ShiftRows()* na criptografia AES é a P-box utilizada no algoritmo. Uma

Tabela 4.1: Velocidade de execução do algoritmo em c

Tamanho (MB)	Tempo de execução (ms)	Velocidade (MB/s)
0.449	1.30	345.4
5,450	16.00	340.0
32.761	94.00	348.5
96.540	312.00	309.4
338.633	1187.00	285.2
712.523	2517.00	284.8
963.453	NA	NA

desvantagem dessa P-box é o fato de ela ser única, ou seja, toda iteração usa a mesma P-box. Com o avanço das técnicas de criptanálise, utilizar a mesmo P-box em cada iteração pode tornar-se uma vulnerabilidade que contribui para a previsibilidade do bloco de dados sendo cifrado.

O algoritmo de permutação proposto nesse trabalho de conclusão de curso corrige essa deficiência, pois ele gera P-boxes de maneira dinâmica através da escolha de chaves. Outra vantagem do algoritmo proposto é o fato dele ser utilizado em um bloco de dados 20x20, contra 4x4 do AES. Como o AES foi proposto em meados do ano 2000, é questão de tempo até que sejam identificados vulnerabilidades em seu código, necessitando a utilização de algoritmos com blocos de dados maiores.

Capítulo 5

Conclusões

Analisando os resultados referentes ao desenvolvimento do servidor de arquivos, apresentado no capítulo 4, pode-se concluir que o objetivo do projeto de desenvolver um servidor web rodando em Linux embarcado foi cumprido. A escolha do servidor LAMP foi essencial para o sucesso desse trabalho, pois o Linux, Apache, MySQL e PHP combinados, formam um servidor robusto para a aplicação que foi proposta. O uso de um *template* responsivo garantiu a qualidade da interface web em diversos navegadores e em dispositivos portáteis e com tamanhos de tela variável. As funções principais do servidor de arquivos (deletar, upload e download) funcionaram corretamente mesmo para arquivos com tamanho da ordem de 700MB. Além disso, o custo do projeto é inferior a assinatura anual de um serviço de armazenamento na nuvem. Conclui-se, portanto, que o servidor web cumpre o objetivo proposto.

O algoritmo de permutação proposto cumpre com o objetivo de gerar permutações de maneira dinâmica, baseadas em chaves definidas pelo usuário. A utilização de diferentes P-boxes durante um algoritmo de criptografia, torna o algoritmo menos previsível e por consequência mais forte. O método como a permutação foi definida, que espalha os elementos de uma mesma linha e coluna, para as linhas e colunas restante é um indicativo que a permutação tem boas propriedades de difusão. Por fim, a proposta de uma P-box dinâmica com o tamanho do bloco bem maior que blocos utilizados nos algoritmos de criptografias atuais, como o AES, torna essa ferramenta interessante para futuros projetos.

Conclui-se, portanto, que o servidor web e o programa que gera permutações ambos cumprem seus objetivos. Através da análise de seus resultados, verificou-se que o servi-

dor web e o permutador são compatíveis, pelo fato de trabalharem em conjunto em arquivos de até 700MB (embora ambas as ferramentas possam ser utilizadas separadamente). Além disso, o permutador tem uma velocidade registrada superior a 280MB/s, ou seja, a etapa de permutação é ágil o suficiente para trabalhar com arquivos grandes e não retardar o tempo de *upload* significativamente.

Trabalhos futuros

Aproveitando o algoritmo que gera P-boxes de tamanho 20x20 que é definido nesse trabalho. Sugere-se a combinação com o algoritmo para criação de S-boxes desenvolvido em [4], cujo resultado é conseguir criar funções bijetoras com boas propriedades criptográficas, como não linearidade, para aplicação em blocos de dados que possuem um número par de elementos. A junção dos dois algoritmos seria através da implementação de etapas iterativas usando S-boxes e P-boxes para gerar confusão e difusão.

Referências

[1] A. Cunha. O que sÃ£o sistemas embarcados? http://files.comunidades.net/mutcom/ARTIGO_SIST_EMB.pdf, Acesso em: 29 de outubro de 2016.

[2] K. Branco. Engenharia de seguranca. http://wiki.icmc.usp.br/images/e/e5/Scc547-101-a05Big.pdf, Acesso em: 20 de outubro de 2016.

[3] W. Stallings. Criptografia e seguranca em rede - capÃtulo 3. http://conteudo.icmc.usp.br/pessoas/otj/SSC0547/Cap03.pdf, Acesso em: 20 de outubro de 2016.

[4] Carlet C. e Tang X. Tang, D. Differentially 4-uniform bijections by permuting the inverse function. https://eprint.iacr.org/2013/639.pdf, Acesso em: 20 de outubro de 2016.

[5] National Institute of Standards and Technology. Federal information processing standards publication 197. http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf, Acesso em: 29 de outubro de 2016.

[6] H. Shinde, G. e Fadewar. Faster rsa algorithm for decryption using chinese remainder theorem. http://www.techscience.com/doi/10.3970/icces.2008.005.255.pdf, Acesso em: 21 de outubro de 2016.

[7] K. Conrad. The chinese remainder theorem. http://www.math.uconn.edu/~kconrad/blurbs/ugradnumthy/crt.pdf, Acesso em: 21 de outubro de 2016.

[8] Raspberry Pi Foundation. Raspberry pi 3 model b. https://www.raspberrypi.org/products/raspberry-pi-3-model-b/, Acesso em: 10 de julho de 2016.

[9] JGraph Ltd. https://draw.io/, Acesso em: 10 de julho de 2016.

[10] Ben. Use usb hard disk & flash drives with your raspberry pi. https://devtidbits.com/2013/03/21/using-usb-external-hard-disk-flash-drives-with-to-your-raspberry-pi/, Acesso em: 21 de outubro de 2016.

[11] Matt. How to mount a usb flash disk on the raspberry pi. http://www.raspberrypi-spy.co.uk/2014/05/how-to-mount-a-usb-flash-disk-on-the-raspberry-pi/, Acesso em: 21 de outubro de 2016.

[12] Apache Project. The number one http server on the internet. https://httpd.apache.org/, Acesso em: 23 de outubro de 2016.

[13] Raspberry Pi Foundation. Lamp web server. https://www.raspberrypi.org/learning/lamp-web-server-with-wordpress/worksheet/, Acesso em: 10 de julho de 2016.

[14] Oracle. Banco de dados mysql. https://www.oracle.com/br/products/mysql/overview/index.html, Acesso em: 23 de outubro de 2016.

[15] PHPeasystep. Php login script tutorial. http://www.phpeasystep.com/phptu/6.html, Acesso em: 23 de outubro de 2016.

[16] Solo Learn. Learn html. https://play.google.com/store/apps/details?id=com.sololearn.htmltrial&hl=pt_BR, Acesso em: 23 de outubro de 2016.

[17] Solo Learn. Learn php. https://play.google.com/store/apps/details?id=com.sololearn.php&hl=pt_BR, Acesso em: 24 de outubro de 2016.

[18] Bootstrap. Getting started. http://getbootstrap.com/getting-started/, Acesso em: 24 de outubro de 2016.

[19] Anli. Bootstrap login forms: 3 free responsive templates. http://azmind.com/2015/04/19/bootstrap-login-forms/, Acesso em: 24 de outubro de 2016.

[20] php.net. readfile. http://php.net/manual/pt_BR/function.readfile.php, Acesso em: 24 de outubro de 2016.

[21] T. Belem. Php: Upload de arquivos. http://www.linhadecodigo.com.br/artigo/3578/php-upload-de-arquivos.aspx, Acesso em: 26 de outubro de 2016.

Apêndice A

Códigos Utilizados no projeto

A.1 Web Site (PHP)

A.1.1 index.php

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="utf-8">
<meta http-equiv="X-UA-Compatible" content="IE=edge">
<meta name="viewport" content="width=device-width, initial-scale=1">
<title>TCC Login Page</title>

<!-- CSS -->

<link rel="stylesheet" href="http://fonts.googleapis.com/css?family=Roboto:400,100,
300,500">
<link rel="stylesheet" href="assets/bootstrap/css/bootstrap.min.css">
<link rel="stylesheet" href="assets/font-awesome/css/font-awesome.min.css">
<link rel="stylesheet" href="assets/css/form-elements.css">
<link rel="stylesheet" href="assets/css/style.css">

<!-- HTML5 Shim and Respond.js IE8 support of HTML5 elements and media queries -->
<!-- WARNING: Respond.js doesn't work if you view the page via file:// -->
```

```
<!--[if lt IE 9]>
<script src="https://oss.maxcdn.com/libs/html5shiv/3.7.0/html5shiv.js"></script>
<script src="https://oss.maxcdn.com/libs/respond.js/1.4.2/respond.min.js"></script>

<![endif]-->

<!-- Favicon and touch icons -->
<link rel="shortcut icon" href="assets/ico/favicon.png">
<link rel="apple-touch-icon-precomposed" sizes="144x144" href=
"assets/ico/apple-touch-icon-144-precomposed.png">
<link rel="apple-touch-icon-precomposed" sizes="114x114" href=
"assets/ico/apple-touch-icon-114-precomposed.png">
<link rel="apple-touch-icon-precomposed" sizes="72x72" href=
"assets/ico/apple-touch-icon-72-precomposed.png">
<link rel="apple-touch-icon-precomposed" href=
"assets/ico/apple-touch-icon-57-precomposed.png">


</head>
<body>

<!-- Top content -->
<div class="top-content">
<div class="inner-bg">
<div class="container">
<div class="row">
<div class="col-sm-8 col-sm-offset-2 text">

<h1><strong>TCC</strong> EESC - USP</h1>
</div>

</div>
```

```
<div class="row">

<div class="col-sm-6 col-sm-offset-3 form-box">
<div class="form-top">
<div class="form-top-left">
<h3>Login</h3>

<p>Entre com seu usuário e senha:</p>
</div>
<div class="form-top-right">
<i class="fa fa-lock"></i>

</div>
</div>

<div class="form-bottom">
<form role="form" action="login.php" method="post" class="login-form">

<div class="form-group">
<label class="sr-only" for="form-username">Username</label>
<input type="text" name="usuario" placeholder="Usuário..."
 class="form-username form-control" id="form-username">

</div>
<div class="form-group">
<label class="sr-only" for="form-password">Senha</label>

<input type="password" name="senha" placeholder="Senha..."
 class="form-password form-control" id="form-password">

</div>
<button type="submit" class="btn">Login!</button>
```

```
</form>
</div>
</div>
</div>
</div>
</div>
</div>

<!-- Javascript -->
<script src="assets/js/jquery-1.11.1.min.js"></script>
<script src="assets/bootstrap/js/bootstrap.min.js"></script>
<script src="assets/js/jquery.backstretch.min.js"></script>
<script src="assets/js/scripts.js"></script>

<!--[if lt IE 10]>
<script src="assets/js/placeholder.js"></script>
<![endif]-->

</body>
</html>
```

A.1.2 login.php

```
<?php


$host = "localhost"; //Nome do Host padrão do Apache
$username = "root"; //usuario Mysql
$password = "roberio"; //senha Mysql
$db_nome = "Usuarios"; //Nome da Base de Dados
$tab_nome = "Tabela"; //nome da Tabela na Base de Dados

//Conectando ao servidor e selecionando a Base de dados
mysql_connect("$host", "$username", "$password")or
```

```
 die("Conecção na BD Falhou". mysql_error());

mysql_select_db("$db_nome")or die("Nao foi possível
 selecionar a BD". mysql_error());



//usuario e senha
$user = $_POST['usuario'];
$pass = $_POST['senha'];

//Protege de MySQL injection
$user = stripslashes($user);
$pass = stripslashes($pass);
$user = mysql_real_escape_string($user);
$pass = mysql_real_escape_string($pass);


$sql = "SELECT * FROM $tab_nome WHERE Usuario='$user' and Senha='$pass'";

$result = mysql_query($sql);

//numero de linhas na tabela da BD
$count = mysql_num_rows($result);

//se o $user e $pass estiverem na tabela, $count > 0
if($count > 0)
{
//inicia sessão
session_start();

//Salva variáveis usuário e senha para identificar a sessão
$_SESSION['login'] = $user;
```

```
$_SESSION[’senha’] = $pass;
$_SESSION[’mensagem’] = "Bem Vindo!";

//entra no site
header("location:site.php");
}

else
{
//destroi sessão
session_destroy();

//limpa variáveis
unset($_SESSION[’login’]);
unset($_SESSION[’senha’]);
unset($_SESSION[’path’]);
unset($_SESSION[’path_back’]);
unset($_SESSION[’mensagem’]);

//volta a página inicial
header("location:index.php");
}
?>
```

A.1.3 logout.php

```
<?php

session_start();
//destroi a sessão
session_destroy();

//limpa ambas variáveis
unset($_SESSION[’login’]);
```

```
unset($_SESSION['senha']);
unset($_SESSION['path']);
unset($_SESSION['path_back']);
unset($_SESSION['mensagem']);

//volta a pagina inicial
header("location:index.php");
?>
```

A.1.4 site.php

```
<!DOCTYPE html>
<html lang="en">

<head>

<meta charset="utf-8">
<meta http-equiv="X-UA-Compatible" content="IE=edge">
<meta name="viewport" content="width=device-width,
 initial-scale=1">

<title>Meus Arquivos</title>
<!-- CSS -->
<link rel="stylesheet" href="http://fonts.googleapis.com/css?family=Roboto:
 400,100,300,500">

<link rel="stylesheet" href="assets/bootstrap/css/bootstrap.min.css">
<link rel="stylesheet" href="assets/font-awesome/css/font-awesome.min.css">
<link rel="stylesheet" href="assets/css/form-elements.css">
<link rel="stylesheet" href="assets/css/style.css">

<!-- HTML5 Shim and Respond.js IE8 support of HTML5 elements andmedia queries -->
<!-- WARNING: Respond.js doesn't work if you view the page via file:// -->
<!--[if lt IE 9]>
```

```
<script src="https://oss.maxcdn.com/libs/html5shiv/3.7.0/html5shiv.js"></script>
<script src="https://oss.maxcdn.com/libs/respond.js/1.4.2/respond.min.js"></script>
<![endif]-->

<!-- Favicon and touch icons -->
<link rel="shortcut icon" href="assets/ico/favicon.png">
<link rel="apple-touch-icon-precomposed" sizes="144x144" href=
"assets/ico/apple-touch-icon-144-precomposed.png">
<link rel="apple-touch-icon-precomposed" sizes="114x114" href=
"assets/ico/apple-touch-icon-114-precomposed.png">
<link rel="apple-touch-icon-precomposed" sizes="72x72" href=
"assets/ico/apple-touch-icon-72-precomposed.png">
<link rel="apple-touch-icon-precomposed" href=
"assets/ico/apple-touch-icon-57-precomposed.png">

</head>
<body>
<?php

session_start();
if( (!isset($_SESSION['login']) == true) or (!isset($_SESSION['senha']) == true) )
{
//destroi a sessão
session_destroy();

//limpa ambas variáveis
unset($_SESSION['login']);
unset($_SESSION['senha']);
unset($_SESSION['path']);
unset($_SESSION['path_back']);
unset($_SESSION['mensagem']);

//volta a pagina inicial
```

```
header("location:index.php");
}

//mensagem de erros para facilitar a programação e debug
ini_set('display_errors', 'On');
error_reporting(E_ALL | E_STRICT);

//caminho da pasta que armazena os arquivos e seu backup.
$path = "/mnt/usb/E010/UPLOAD/";
$path_back = "/mnt/usb/E010/BKG/";

//guarda os caminhos para uso global
$_SESSION['path'] = $path;
$_SESSION['path_back'] = $path_back;

//$files é vetor com nome dos arquivos em path, $total
 é o numero de arquivos
$files = scandir($path);
$total = count($files);
?>

<!-- Top content -->
<div class="top-content">
<div class="col-sm-8 col-sm-offset-2 text" style=
"padding: 20px;font-size:20px" >
<?php echo $_SESSION['mensagem'];
$_SESSION['mensagem']="";
?>
</div>
<div class="inner-bg">
<div class="col-sm-8 col-sm-offset-3 text">

<!-- Botão de Sair-->
```

```
<a href="logout.php" style="float: right;margin-right:
 10px"><button type="button" class="btn btn-danger">
Sair</button></a>
</div>

<div class="container">
<div class="row">
<div class="col-sm-4 col-sm-offset-4 text">
<!-- form de upload -->

<form action=upload.php method="POST" enctype="multipart/form-data">

<label>Upload máximo 1GB!</label>
<input type="hidden" name="MAX_FILE_SIZE" value="1048576000" />
<input type="file" name="arquivo_up" />
<input type="submit" value="Enviar" class="btn btn-lgbtn-primary"/>



</form>
</div>
</div>
<div class="table-responsive">
<table class="table table">
<thead>
<tr>
<th>Nome</th>
<th class="col-md-1">Tamanho</th>
<th class="col-md-1">Deletar</th>
</tr>
</thead>
<tbody>

<?php
```

```
//cria links para todos os arquivos presentes
for($x = 2; $x < $total; $x++)
{
?>
<tr>
<td><?php echo "<a href='download.php/?arquivo=".$files[$x]."'>"
.$files[$x]."</a>"; ?></td>
<td><?php if(filesize($path.$files[$x])<=1024){
echo filesize($path.$files[$x])."B";}
elseif(filesize($path.$files[$x])<=1048576){
echo round(filesize($path.$files[$x])/1024,2)."KB";}
else{
echo round(filesize($path.$files[$x])/1048576,2)."MB";}
?></td>

<td><?php echo "<a href='deleta.php/?arquivo=".$files[$x]."'>Deletar</a>";
 ?></td>

</tr>
<?php
}
?>

</tbody>
</table>
</div>
</div>
</div>
</div>

<!-- Javascript -->
<script src="assets/js/jquery-1.11.1.min.js"></script>
<script src="assets/bootstrap/js/bootstrap.min.js"></script>
```

```
<script src="assets/js/jquery.backstretch.min.js"></script>
<script src="assets/js/scripts.js"></script>

<!--[if lt IE 10]>
<script src="assets/js/placeholder.js"></script>
<![endif]-->
</body>
</html>
```

A.1.5 upload.php

```
<?php
session_start();

if( (!isset($_SESSION['login']) == true) or (!isset($_SESSION['senha']) == true) )
{
//destroi a sessão
session_destroy();

//limpa ambas variáveis
unset($_SESSION['login']);
unset($_SESSION['senha']);
unset($_SESSION['path']);
unset($_SESSION['path_back']);
unset($_SESSION['mensagem']);

//volta a pagina inicial
header("location:index.php");
}

//mensagens de erro para debug
ini_set('display_errors', 'On');
error_reporting(E_ALL | E_STRICT);
```

```
//Pasta onde o arquivo vai ser salvo e seu backup
$path = $_SESSION['path'];
$path_back = $_SESSION['path_back'];

//Array com os tipos de erros de upload do PHP
$Erro[0] = "Não houve erro";
$Erro[1] = "O arquivo no upload é maior do que o limite do PHP";
$Erro[2] = "O arquivo ultrapassa o limite de tamanho especifiado no HTML";
$Erro[3] = "O upload do arquivo foi feito parcialmente";
$Erro[4] = "Não foi feito o upload do arquivo";

//Verifica se houve algum erro com o upload. Se sim, exibe a mensagem do erro
if($_FILES['arquivo_up']['error'] != 0)
{
die("Não foi possível fazer o upload, erro:
 ".$Erro[$_FILES['arquivo_up']['error']]);
exit; // Para a execução do script
}

//nome do arquivo a ser upado
$nome_final = basename($_FILES['arquivo_up']['name']);

//$ files é vetor com nome dos arquivos em path, $total é o numero de arquivos
$files = scandir($path);
$total = count($files);

for($x = 2; $x < $total; $x++)
{
if($nome_final == $files[$x])
{
$_SESSION['mensagem']="Arquivo ".$files[$x]." já existente!";
header("Location:site.php");
exit;
```

```
}
}

//move o arquivo para path
$check = move_uploaded_file($_FILES['arquivo_up']['tmp_name'], $path.$nome_final);

//Verifica se o arquivo upado foi movido para pasta escolhida
if($check)
{
$check = copy($path.$nome_final, $path_back.$nome_final);
if($check)
{
//upload e backup realizados com sucesso.
$_SESSION['mensagem']="Arquivo ".$nome_final." upado com sucesso!";
header("Location:site.php");
}

else
{
//Não foi possível fazer o upload, verifique o dispositivo de salvemento ou a pasta

echo "Arquivo upado, porém não foi possível enviar o arquivo de backup ".$nome_final.
}
}
else
{
//Não foi possível fazer o upload, verifique o dispositivo de salvemento ou a pasta

echo "Não foi possível upar o arquivo ".$nome_final.", tente novamente mais tarde.";
}
?>
```

A.1.6 download.php

```
<?php

session_start();

if( (!isset($_SESSION['login']) == true) or
(!isset($_SESSION['senha']) == true) )
{
//destroi a sessão
session_destroy();

//limpa ambas variáveis
unset($_SESSION['login']);
unset($_SESSION['senha']);
unset($_SESSION['path']);
unset($_SESSION['path_back']);
unset($_SESSION['mensagem']);

//volta a pagina inicial
header("location:index.php");
}

//mensagem de erros para facilitar a programação e debug
ini_set('display_errors', 'On');
error_reporting(E_ALL | E_STRICT);

//seta limite de tempo de download em infinito
set_time_limit(0);

//pasta com arquivos a serem deletados
$path = $_SESSION['path'];
$arquivo = $_GET['arquivo'];
$arquivo = $path.$arquivo;
```

```
$tamanho = filesize($arquivo);

//headers para tranferência de arquivos
header('Content-Description: File Transfer');
header('Content-Type: application/octet-stream');
header('Content-Transfer-Encoding: Binary');
header('Content-Disposition: attachment; filename="'.basename($arquivo).'"');
header('Expires: 0');
header('Cache-Control: must-revalidate, post-check=0, pre-check=0');
header('Cache-Control: private', false);
header('Pragma: public');
header('Content-Length:'.$tamanho);

readfile($arquivo);
exit;
?>
```

A.1.7 deleta.php

```
<center>
<br><br>
<?php

session_start();

if( (!isset($_SESSION['login']) == true) or (!isset($_SESSION['senha']) == true) )
{
//destroi a sessão
session_destroy();

//limpa ambas variáveis
unset($_SESSION['login']);
unset($_SESSION['senha']);
unset($_SESSION['path']);
```

```
unset($_SESSION['path_back']);
unset($_SESSION['mensagem']);

//volta a pagina inicial
header("location:index.php");
}

//mensagem de erros para facilitar a programação e debug
ini_set('display_errors', 'On');
error_reporting(E_ALL | E_STRICT);

//Pasta onde o arquivo vai ser salvo e seu backup
$path = $_SESSION['path'];
$path_back = $_SESSION['path_back'];

//deleta arquivo
$file = $_GET['arquivo'];
$nome = $path.$file;

if(unlink($nome))
{
//deleta arquivo backup
$nome = $path_back.$file;

if(unlink($nome))
{
echo $file." deletado com sucesso!";
}
else
{
echo "Arquivo de backup ".$file." não conseguiu ser deletado!";
exit;
}
```

```
}
else
{
echo "Arquivo ".$file." não conseguiu ser deletado!";
exit;
}
?>

<br>
<form>
<input type="button" value="Voltar" onClick="window.history.go(-1);return true;">
</form>
</center>
```

A.2 Algoritmo P-box

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>

 //recebe a cifra de entrada com 400bytes e retorna a cifra permutada
 //com o método do Teorema Chinês do Restos
int *chines_cryp(int *entrada, long long int tamanho)
{
long long int i,j,n;
int *saida = (int *)malloc((tamanho)*sizeof(int));

//n é o menor lado da permutation box
n = floor((1+sqrt(1+4*tamanho))/2);

for(i=0;i<tamanho;i++)
{
```

```
j = (i%n)+(i%(n+1))*n;
if(j>=tamanho)
{
j = (j%n)+(j%(n+1))*n;
saida[j] = entrada[i];
}
else
{
saida[j] = entrada[i];
}
}
return saida;
}

//recebe a cifra permutada com 400bytes e retorna a despermutada
//com o método do Teorema Chinês do Restos
int *chines_decryp(int *entrada, long long int tamanho)
{
long long int i,j,n;
int *saida = (int *)malloc((tamanho)*sizeof(int));

//n é o menor lado da permutation box
n = floor((1+sqrt(1+4*tamanho))/2);

for(i=0;i<tamanho;i++)
{
j = (i%n)+(i%(n+1))*n;
if(j>=tamanho)
{
j = (j%n)+(j%(n+1))*n;
saida[i] = entrada[j];
}
else
```

```
{
saida[i] = entrada[j];
}
}
return saida;
}

//função que retorna a permutação de um vetor, utilizando chave x
int *permuta_vetor(long long int x)
{
int *perm = (int *)malloc((20)*sizeof(int));
long long int fatorial[20]= {1,1,2,6,24,120,720,5040,40320,362880,3628800,39916800,
479001600,6227020800,87178291200,1307674368000,
20922789888000,355687428096000,6402373705728000,
121645100408832000}; //0! até 19!
int i, j[20],k;

for(i=19;i>-1;i--)
{
perm[i] = -1;
j[i] = x/fatorial[i];
x = x%fatorial[i];
}

for(i=0;i<20;i++) //perm é o vetor permutado
{
if(perm[j[i]]==-1)
{
perm[j[i]]=i;
}
else
{
for(k=i;k>j[i];k--)
```

```
{
perm[k]=perm[k-1];
}
perm[j[i]]=i;
}
}
return perm;
}

//função que aplica as permutações nas linhas e colunas de uma cifra
//utilizando os vetores de eprmutação horizontal[20] e vertical[20]
int *permuta_h_v(int *entrada,int horizontal[20],int vertical[20])
{
int *saida = (int *)malloc((400)*sizeof(int));
int i;

for(i=0;i<400;i++)
{
saida[i] = entrada[horizontal[i%20]+vertical[i/20]*20];
}
return saida;
}

 //função que transpoe uma matriz 20x20.
int *transpoe(int *entrada)
{
int *saida = (int *)malloc((400)*sizeof(int));
int i,j;

for(i=0;i<20;i++)
{
for(j=0;j<20;j++)
{
```

```
saida[i*20+j]=entrada[j*20+i];
}
}
return saida;
}

//função que permuta cada bloco de 400bytes do arquivo de entrada, //devolvendo a saí
char *p_box(char *entrada, char *saida, long long int tamanho, long long int *chave)
{
int *perm1,*perm2,*perm3,*perm4,i;
int *cifra = (int *)malloc((400)*sizeof(int));
long long int j,k;

perm1 = permuta_vetor(chave[0]);
perm2 = permuta_vetor(chave[1]);
perm3 = permuta_vetor(chave[2]);
perm4 = permuta_vetor(chave[3]);

for(i=0;i<400;i++)
{
cifra[i] = i;
}

cifra = chines_cryp(cifra,400);
cifra = permuta_h_v(cifra,perm1,perm2);
cifra = transpoe(cifra);
cifra = chines_cryp(cifra,400);
cifra = permuta_h_v(cifra,perm3,perm4);

j = tamanho/400;

for(k=0;k<j;k++)
{
```

```
for(i=0;i<400;i++)
{
saida[400*k+i]=entrada[400*k+cifra[i]];
}
}
return saida;
}

//função que despermuta cada bloco de 400bytes do arquivo de entrada, //devolvendo a
char *p_box_inv(char *entrada, char *saida, long long int tamanho,
 long long int *chave)
{
int *perm1,*perm2,*perm3,*perm4,i;
int *cifra = (int *)malloc((400)*sizeof(int));
long long int j,k;

perm1 = permuta_vetor(chave[0]);
perm2 = permuta_vetor(chave[1]);
perm3 = permuta_vetor(chave[2]);
perm4 = permuta_vetor(chave[3]);

for(i=0;i<400;i++)
{
cifra[i] = i;
}

cifra = chines_cryp(cifra,400);
cifra = permuta_h_v(cifra,perm1,perm2);
cifra = transpoe(cifra);
cifra = chines_cryp(cifra,400);
cifra = permuta_h_v(cifra,perm3,perm4);

j = tamanho/400;
```

```
for(k=0;k<j;k++)
{
for(i=0;i<400;i++)
{
saida[400*k+cifra[i]]=entrada[400*k+i];
}
}
return saida;
}

// função que mostra a P-box
void mostra_pbox(long long int *chave)
{
int *perm1,*perm2,*perm3,*perm4,i,j;
int *cifra = (int *)malloc((400)*sizeof(int));

perm1 = permuta_vetor(chave[0]);
perm2 = permuta_vetor(chave[1]);
perm3 = permuta_vetor(chave[2]);
perm4 = permuta_vetor(chave[3]);

for(i=0;i<400;i++)
{
cifra[i] = i;
}

cifra = chines_cryp(cifra,400);
cifra = permuta_h_v(cifra,perm1,perm2);
cifra = transpoe(cifra);
cifra = chines_cryp(cifra,400);
cifra = permuta_h_v(cifra,perm3,perm4);
```

```
for(i=0;i<20;i++)
{
printf("\n");
for(j=0;j<20;j++)
{
printf(" %03d ",cifra[20*i+j]);
}
}
}

int main()
{
FILE *file;
char *entrada, *saida, *arquivo=(char *)malloc((500)*sizeof(char));
//char tem tamanho de 1 byte
long long int tamanho, *chave=(long long int *)malloc((4)*sizeof(long long int));
int menu;

while(1) //menu do programa
{
printf(" ***** Gerador de Permutation Boxes *****\n\n");
printf(" Escolha uma opcao:\n\n 1 - P-BOX\n 2 - P-BOX inversa\n
 3 - Mostra P-BOX\n 4 - Sair\n");
scanf("%d",&menu);
fflush(stdin);

switch(menu)
{
case 1:
printf("\nDigite nome e endereco completo do arquivo de entrada:\n");
gets(arquivo);
fflush(stdin);
```

```
file = fopen(arquivo, "rb");    //Abre o arquivo em modo binario
if(file==NULL)
{
printf("\nArquivo nao encontrado, pressione enter para voltar ao menu:\n");
gets(arquivo);
fflush(stdin);
system("cls");
break;
}
fseek(file, 0, SEEK_END);     //calcula o tamanho do arquivo
tamanho = ftell(file);
rewind(file);

printf("\nDigite a chave 1, de 0 ate 2432902008176639999:\n");
scanf("%lld",&chave[0]);
fflush(stdin);
printf("Digite a chave 2, de 0 ate 2432902008176639999:\n");
scanf("%lld",&chave[1]);
fflush(stdin);
printf("Digite a chave 3, de 0 ate 2432902008176639999:\n");
scanf("%lld",&chave[2]);
fflush(stdin);
printf("Digite a chave 4, de 0 ate 2432902008176639999:\n");
scanf("%lld",&chave[3]);
fflush(stdin);

//aloca memoria para os arquivos de entrada e permutados.
entrada = (char *)malloc((tamanho+1)*sizeof(char));
saida = (char *)malloc((tamanho+1)*sizeof(char));

fread(entrada, tamanho, 1, file); // Read in the entire file
fclose(file); // Close the file
```

```
saida = p_box(entrada,saida,tamanho,chave);

printf("\nDigite nome e endereco completo do arquivo de saida:\n");
gets(arquivo);
fflush(stdin);

file = fopen(arquivo, "w+b");
fwrite(saida, tamanho, 1, file);
fclose(file);

printf("\nArquivo permutado gerado com sucesso,
 pressione enter para voltar ao menu:\n");
gets(arquivo);
fflush(stdin);
system("cls");
break;

case 2:
printf("\nDigite nome e endereco completo do arquivo de entrada:\n");
gets(arquivo);
fflush(stdin);

file = fopen(arquivo, "rb");
if(file==NULL)
{
printf("\nArquivo nao encontrado, pressione enter para voltar ao menu:\n");
gets(arquivo);
fflush(stdin);
system("cls");
break;
}
fseek(file, 0, SEEK_END);    //calcula o tamanho do arquivo
tamanho = ftell(file);
```

```
rewind(file);

printf("\nDigite a chave 1, de 0 ate 2432902008176639999:\n");
scanf("%lld",&chave[0]);
fflush(stdin);
printf("Digite a chave 2, de 0 ate 2432902008176639999:\n");
scanf("%lld",&chave[1]);
fflush(stdin);
printf("Digite a chave 3, de 0 ate 2432902008176639999:\n");
scanf("%lld",&chave[2]);
fflush(stdin);
printf("Digite a chave 4, de 0 ate 2432902008176639999:\n");
scanf("%lld",&chave[3]);
fflush(stdin);

entrada = (char *)malloc((tamanho+1)*sizeof(char));
saida = (char *)malloc((tamanho+1)*sizeof(char));

fread(entrada, tamanho, 1, file);
fclose(file);

saida = p_box_inv(entrada,saida,tamanho,chave);

printf("\nDigite nome e endereco completo do arquivo de saida:\n");
gets(arquivo);
fflush(stdin);

file = fopen(arquivo, "w+b");
fwrite(saida, tamanho, 1, file);
fclose(file);

printf("\nArquivo despermutado gerado com sucesso,
 pressione enter para voltar ao menu:\n");
```

```
gets(arquivo);
fflush(stdin);
system("cls");
break;

case 3:
printf("\nDigite a chave 1, de 0 ate 2432902008176639999:\n");
scanf("%lld",&chave[0]);
fflush(stdin);
printf("Digite a chave 2, de 0 ate 2432902008176639999:\n");
scanf("%lld",&chave[1]);
fflush(stdin);
printf("Digite a chave 3, de 0 ate 2432902008176639999:\n");
scanf("%lld",&chave[2]);
fflush(stdin);
printf("Digite a chave 4, de 0 ate 2432902008176639999:\n");
scanf("%lld",&chave[3]);
fflush(stdin);

mostra_pbox(chave);
printf("\n\nPressione enter para voltar ao menu:\n");
gets(arquivo);
fflush(stdin);
system("cls");
break;

case 4:
return 0;
}   //fim do SWITCH
}   //fim do while(1)

}
```