UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANA
DEPARTAMENTO ACADÊMICO DE ELETRÔNICA
CURSO DE ESPECIALIZACÃO EM CONFIGURAÇÃO E GERENCIAMENTO DE SERVIDORES E EQUIPAMENTOS DE REDES

JOÃO MARCOS MORETTI PELLISSARI

MECANISMO DE AUTENTICAÇÃO PARA LIBERAR ACESSO AO SERVIDOR LINUX - UM ESTUDO DE CASO

MONOGRAFIA

CURITIBA
2012

JOÃO MARCOS MORETTI PELLISSARI

MECANISMO DE AUTENTICAÇÃO PARA LIBERAR ACESSO AO SERVIDOR LINUX - UM ESTUDO DE CASO

Monografia apresentada como requisito parcial para obtenção do grau de especialista em Configuração e Gerenciamento de Servidores e Equipamentos de Redes, do Departamento Acadêmico de Eletrônica da Universidade Tecnológica Federal do Paraná.
Orientador: Prof. MSc. Juliano Mello

CURITIBA
2012

Á minha esposa Daniele de Araujo Pellissari, que sem seu apoio incondicional eu não seria a pessoa que sou hoje.

RESUMO

MORETTI PELLISSARI, João Marcos. MECANISMO DE AUTENTICAÇÃO PARA LIBERAR ACESSO AO SERVIDOR LINUX - UM ESTUDO DE CASO. 2012. 37 f. Monografia (Especialização em Gerenciamento de Redes) – Programa de Pós-Graduação em Tecnologia, Universidade Tecnológica Federal do Paraná. Curitiba, 2012.

O objetivo deste trabalho é estudar e implementar um mecanismo de autenticação antes que um usuário tenha acesso á portas liberadas do firewall de uma empresa.
A pesquisa apresenta definições sobre segurança de redes, juntamente com uma implementação que melhora a segurança das aplicações que necessitam de acesso remoto, utilizando tecnologias livres que possuem grande aceitação no mercado.

Palavras-chave: IPtables. Segurança em Firewall. Autenticação.

ABSTRACT

MORETTI PELLISSARI, João Marcos. **AUTHENTICATION MECHANISM FOR RELEASE LINUX SERVER ACCESS - A CASE STUDY.** 2012. 27 f. Essay (Graduate Certificate in Networking and Systems Administration) - Graduate Programs in Technology, Federal Technological University of Paraná. Curitiba, 2012.

This work has a object the implementing method to authenticate user before this user access a open port on firewall.
The research show definitions of security network, together with implementation that improves the security with remote access applications.
All work using tecnologies open-souce used by many companies.

Keywords: IPtables. Security Firewall. authenticate.

.

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

1. INTRODUÇÃO

1.1 TEMA

O ambiente coorporativo necessita que seus funcionários, clientes e fornecedores possam acessar remotamente suas informações internas, envolvendo os mais diversos sistemas das mais diferentes organizações. Para isso inúmeras ferramentas de acesso remoto são criadas utilizando uma ampla variedade de tecnologias. Onde cada empresa estuda a melhor forma para disponibilizar o acesso remoto a seus sistemas, implementando tecnologias já utilizadas ou desenvolvendo sua própria forma de conexão.

Uma das maiores dificuldades é prover a disponibilidade, facilidade de acesso e garantir a segurança sobre as informações trafegadas.

Grandes organizações tem um orçamento para TI e dentro deste orçamento, um valor destinado à segurança de suas redes. Geralmente elas tem uma documentação e um projeto de segurança avançado, que é constantemente atualizado. Em pequenas organizações o fator segurança dos dados frequentemente passa despercebido, muitas vezes devido a empresa não dispor de uma equipe de TI ou reserva no orçamento para este fim.

1.2 DELIMITAÇÃO DA PESQUISA

Será implementado um mecanismo de autenticação em servidores Linux que autoriza o acesso do usuário através de um servidor *firewall*. Após sua autenticação seu IP verdadeiro é registrado no *firewall* e seu acesso a alguma porta específica é liberado por um período de tempo previamente configurado pelo administrador.

Este trabalho apresentará inicialmente alguns conceitos de segurança de redes, que juntos implementados podem garantir um nível satisfatório de segurança da informação para pequenas corporações.

A implementação de alguns serviços que em conjunto irão fornecer a autenticação do usuário é o ponto principal deste trabalho. Ele visa mostrar que varias técnicas utilizadas em conjunto podem promover um nível de segurança eficaz para pequenas organizações.

Existem diversas tecnologias já desenvolvidas para fornecer o acesso remoto, como a VPN que seria uma forma mais segura de implementar segurança para os quesitos abordados no trabalho, porém ela exige maior conhecimento da equipe de TI, maior manutenção e pode ocasionar dificuldades de acesso com usuários inexperientes, elevando o

custo de gestão, aliada a possível dificuldade no acesso, assim impossibilitando a adoção desta solução.

1.3 PROBLEMA

Com o crescimento da dependência de acesso dos colaboradores, clientes, fornecedores e parceiros a sistemas que há muito pouco tempo atrás eram apenas internos e geralmente acessados sempre de um mesmo computador, juntamente com a necessidade de mobilidade e portabilidade, muitas maneiras de acesso remoto surgiram, permitindo que o gerente de TI estude uma vasta variedade de ferramentas para fornecer o acesso remoto autorizado em suas redes. Alguns exemplos são: VPN, *Terminal Server*, Sistemas WEB, Links dedicados, entre outros.

Inicialmente apenas grandes organizações tinham tal necessidade, e estas organizações em tese estão providas de uma excelente política de segurança fortemente documentada, testada e ampla. A realidade é que empresas dos mais diversos tamanhos compartilham aplicações pela internet, um ambiente que não é seguro, e devido a isso surge a necessidade também para estas organizações de mecanismos de segurança como *firewalls*, IDS entre outros. Geralmente quando *firewalls* são instalados algumas portas para o acesso remoto são abertas, por exemplo site web, Terminal Server, ou outra aplicação, sendo que qualquer usuário com má intenção pode tentar acessar este recurso e prover algum malefício para a organização, como o uso indevido de informações ou mesmo apenas a negação do serviço, que consiste em deixar indisponível a aplicação para usuários válidos.

Muitas dessas aplicações apresentam sua forma de autenticação própria, geralmente solicitando usuário e senha. Utilizando o conceito de segurança em camadas este trabalho propõe uma autenticação antes de acessar o sistema remoto propriamente dito, mesmo ele tendo algum dispositivo de segurança.

1.4 OBJETIVOS

A seguir, serão apresentados os objetivos geral e específicos, que se pretende atingir com este projeto de pesquisa.

1.4.1 Objetivo Geral

Estudar e implementar uma ferramenta de autenticação simples em ambiente web para que funcionários remotos possam acessar portas específicas do firewall e assim conseguir conectar na rede interna da empresa.

1.4.2 Objetivos Específicos

Os objetivos específicos são:

- Configurar um *firewall* com *iptables*;
- Criar uma aplicação web que permita acesso autenticação e consiga interagir com o *iptables*;
- Após autenticação de usuário e senha devemos permitir acesso a uma ou várias portas do *firewall* que estão bloqueadas para outros endereços IPS.

1.5 JUSTIFICATIVA

A segurança está relacionada à necessidade de proteção contra o acesso ou manipulação, intencional ou não, de informações confidenciais por elementos não autorizados, e a utilização não autorizada do computador ou de seus dispositivos periféricos. A necessidade de proteção deve ser definida em termos das possíveis ameaças e riscos e dos objetivos de uma organização, formalizados nos termos de uma política de segurança. (SOARES; LEMOS e COLCHER,1995, p.448):

Empresas com orçamento de TI limitados muitas vezes necessitam melhorar a segurança do acesso remoto de seus funcionários, clientes e parceiros. Ao final deste projeto elas terão uma orientação de como incrementar a segurança de um *firewall* baseado em *iptables* sem investimento em licenças de software ou hardware especifico, pois todos os temas abordados neste projeto serão de fácil implementação para empresas que já possuam um *firewall*, ou que ainda vão implementa-lo.

Este projeto não é uma solução definitiva para segurança e sim a proposta de uma nova etapa de autenticação para obter acesso a um recurso específico. Cada etapa de segurança aplicada dificulta um usuário com intenção maléfica de chegar ao seu êxito.

Um exemplo é o acesso remoto via Terminal Server, que por si só já apresenta a autenticação Windows implementada. Abaixo apresentamos uma tela com o relatório de

acesso, onde um usuário inválido tenta acessar o servidor de terminal sem conhecer um usuário e senha. Quando simplesmente disponibilizamos o acesso via Terminal Server para a internet, um invasor pode tentar um ataque de força bruta para descobrir a senha, ou mesmo pode utilizar da engenharia social descobrindo o nome dos funcionários e informações que podem ajudar ele na descoberta da senha.

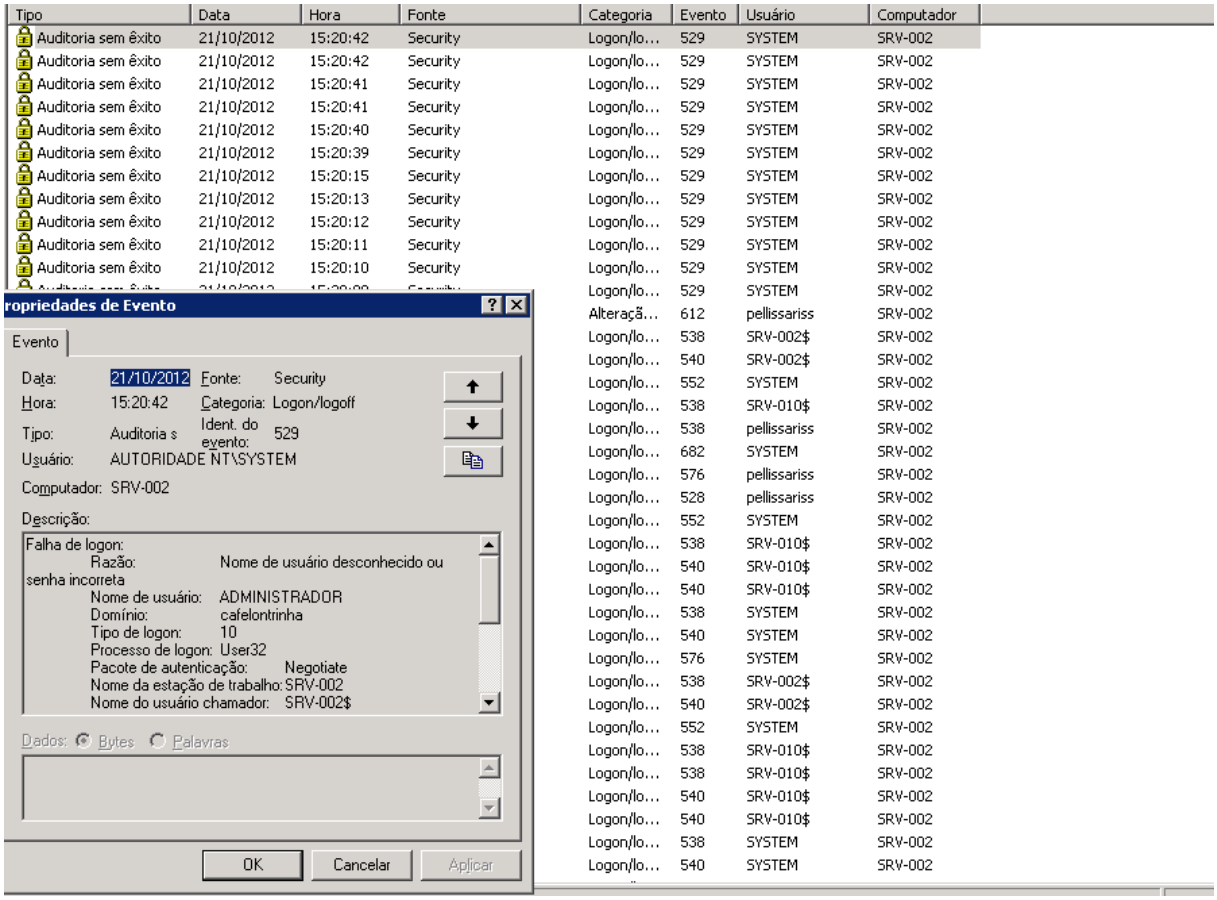

Figura 1 – Log auditoria do Windows Server [Autoria própria].

A figura 1 demostra os eventos de falha de autenticação por um usuário externo no servidor de terminal, quando o sistema de *login* para acessar aporta do *iptables* estiver em vigor, ele terá primeiro que autenticar no *firewall* para depois conseguir acesso ao servidor de terminal, o que dificulta o trabalho do usuário mal intencionado.

1.6 PROCEDIMENTOS METODOLÓGICOS

Para o desenvolvimento deste projeto, utilizar-se-á referências bibliográficas sobre os assuntos pressupostos, bases para o desenvolvimento do trabalho, como Linux, *iptables*, Apache, PHP e segurança de redes.

O estudo utilizará a implementação de um *firewall iptables* como aplicativo principal, ele será o responsável por fornecer a segurança necessária. Um website em PHP tem por objetivo capturar as credenciais do usuário: *login* e senha, e definir quais portas ele tem permissão de acesso. Após isso através de sua integração com o *iptables*, o site deve adicionar uma entrada no *firewall* permitindo estas portas para o IP que fez a autenticação.

O estudo avançado de alguns pacotes utilizados como *iptables*, Apache e PHP não fazem parte do escopo deste trabalho, sendo que existem inúmeros artigos tratando de forma específica destes temas.

1.7 FUNDAMENTAÇÃO TEÓRICA

Para descrever os conceitos de segurança da informação, será utilizada uma revisão de literatura, baseada nas obras de Dias, 2000, Sêmoba, 2003, Nakamura e Geus, 2003. E ainda uma revisão na ISSO/IEC 17799, 2005. Estas biografias são essenciais para explicar a importância da segurança da informação e seu alto valor agregado.

A fase prática deste trabalho será apoiada pelas publicações de Souza, 2004, Machado, 2010 para a implementação do servidor Apache. Esquivel, 2006 e Favero 2007 para a explicação e implementação de *firewall iptables*.

1.8 ESTRUTURA

Esta monografia é estruturada em 5 capítulos, que tem como objetivo de juntos fornecer um guia para a implementação deste sistema de autenticação para liberar portas do *firewall*. No capítulo 1, capítulo introdutório a seguinte estrutura é formulada tendo início com: i) tema de pesquisa; ii) apresentação do problema; iii) objetivos; iv) justificativa; v) procedimentos metodológicos; vi) fundamentação teoria; vii) estrutura.

Para justificar a necessidade de segurança da informação o capitulo 2, refere-se a ela exemplificando seu uso, e trazendo uma justificativa baseada em publicações científicas como

justificativa para a implementação deste trabalho. O Capitulo 3 está explicando os *softwares* LINUX que serão utilizados para o desenvolvimento desta monografia. A parte mais importante da mesma concentra-se no capitulo 4, que mostra a implementação de um sistema que permite a autenticação do usuário antes dele acessar um NAT do *firewall*.

No capítulo 5 é apresentada a conclusão da monografia e suas considerações futuras, descrevendo os resultados, aplicabilidade e utilização desta forma de autenticação.

2. SEGURANÇA DA INFORMAÇÃO

A historia da humanidade mostra que o homem, sempre procurou defender a informação, desde a época que a informação era passada de pai para filhos. Mais recentemente podemos exemplificar as mensagens criptografas entre os exercidos de um pais em guerra, até chegarmos a informações empresariais confidencias de uma fórmula como, por exemplo, a da Coca Cola.

A informação é um ativo que, como qualquer outro ativo importante para os negócios, tem um valor para a organização e consequentemente necessita ser adequadamente protegida (NBR 17999, 2003). Na sociedade da informação, a informação é o principal patrimônio da empresa e está sob constante risco (Dias, 2000). A informação representa a inteligência competitiva dos negócios e é reconhecida como ativo crítico para a continuidade operacional e saúde da empresa (Sêmola, 2003). A informação e o conhecimento serão os diferenciais das empresas e dos profissionais que pretendem destacar-se no mercado e manter a sua competitividade (Rezende e Abreu, 2000).

A segurança da informação visa reduzir os riscos de vazamento de informação, fraude e sabotagens e outras formas de ameaça para uma organização.

A necessidade de segurança é um fato que vem transcendendo o limite de produtividade e da funcionalidade. Enquanto a velocidade e a eficiência em todos os processos de negócios significam uma vantagem competitiva, a falta de segurança nos meios que habilitam a velocidade e a eficiência pode resultar em grandes prejuízos e a falta de novas oportunidades de negócios. (NAKAMURA e GEUS, 2003).

De acordo com a norma ISO/IEC 17799, podemos definir a segurança da informação da seguinte maneira: Segurança da informação é a proteção da informação de vários tipos de ameaças para garantir a continuidade do negócio, minimizar o risco ao negócio, maximizar o retorno sobre os investimentos e as oportunidades de negócio. (Norma ISO/IEC 17799, 2005, p.9).

Segundo Caruso (1999), o bem de maior valor para uma empresa pode não ser exatamente o que é produzido em sua linha de produção, mas sim as informações relacionadas a este bem de consumo ou serviço.

Como citado acima é indiscutível o valor e a importância da informação, e por isso, surge a necessidade de protegê-la.

Para isso discorreremos o principio de segurança da informação:

Segundo Albuquerque (2002) e Krause (1999) há três princípios básicos para garantir a segurança da informação:

• Confidencialidade. A informação somente pode ser acessada por pessoas explicitamente autorizadas. É a proteção de sistemas de informação para impedir que pessoas não autorizadas tenham acesso.

• Disponibilidade. A informação deve estar disponível no momento em que a mesma for necessária.

• Integridade. A informação deve ser recuperada em sua forma original (no momento em que foi armazenada). É a proteção dos dados ou informações contra modificações intencionais ou acidentais não-autorizadas. (LAUREANO e MORAES, 2005)

Para proteger nossas informações existem varias tecnologias ou conjunto de tecnologias aplicadas que juntas aplicadas nos fornecem a proteção necessária para atender os três princípios descritos acima.

É na politica de segurança da empresa que definimos como vamos tratar cada um dos princípios básicos da segurança da informação, neste momento são avaliadas as tecnologias que serão empregadas (*firewall*, IDS, politicas de senha, politicas de acesso a recursos, acesso físico entre outros).

Sabemos que o processo de segurança da informação nunca está finalizado, novas ameaças e necessidades de acesso surgem rapidamente, para isso o processo de segurança deve ser constantemente revisado, testado e consequentemente aperfeiçoado.

As redes de computadores, e principalmente a internet mudou a forma de utilização dos sistemas de informação. As possibilidades e oportunidades de utilização são muito mais amplas que em sistemas fechados, assim como os riscos à privacidade e integridade da informação. Portanto, é muito importante que mecanismos de segurança de sistemas de informação sejam projetados de maneira a prevenir acessos não autorizados aos recursos e dados destes sistemas (Laureano, 2004).

Dependendo do tipo de informação e da necessidade de confiabilidade, autenticidade, integridade e disponibilidade, é determinada a ferramenta ou o conjunto delas para garantir que os requisitos sejam cumpridos.

Segundo a norma NBR ISO/IETC 17799:2005 o objetivo da política de segurança é “Prover uma orientação e apoio da direção a segurança da informação de acordo com os requisitos do negocio e com as lei e regulamentações relevantes.”

Em resumo: A segurança da informação é a proteção dos sistemas de informação contra a negação de serviço a usuários autorizados, assim como contra a intrusão, e a modificação não-autorizada de dados ou informações, armazenados, em processamento ou em trânsito, abrangendo a segurança dos recursos humanos, da documentação e do material, das áreas e instalações das comunicações e computacional, assim como as destinadas a prevenir, detectar, deter e documentar eventuais ameaças a seu desenvolvimento (NBR 17999, 2003; Dias, 2000; Wadlow, 2000; Krause e Tipton, 1999).

3. APLICATIVOS UTILIZADOS

3.1 APACHE

O Servidor Apache é responsável pela distribuição de documentos HTTP e HTTPS, sendo o servidor web um *software* que controla a recuperação e disponibilização dos arquivos em formato de hypertexto em um sistema de arquivos.

Segundo (SOUZA, 2004), o Apache *Web Server* é um Servidor *Web*, open source, multi-plataforma (BSD, GNU/Linux, UNIX systems, Microsoft Windows e outras plataformas) mais utilizado em todo mundo. As principais características do Apache são flexibilidade, altamente configurável, robustez, escalabilidade, pode ser configurado para diferentes funções e é composto de módulos separados onde cada um implementa uma característica diferente.

Segundo (MACHADO,2010) Para garantir segurança nas transações HTTP, o servidor dispõe de um módulo chamado mod_ssl, o qual adiciona a capacidade do servidor atender requisições utilizando o protocolo HTTPS (*HyperText Transfer Protocol Secure*). Este protocolo utiliza uma camada SSL (*Secure Sockets Layer*) para criptografar todos os dados transferidos entre o cliente e o servidor, provendo maior grau de segurança, confidencialidade e confiabilidade dos dados. A camada SSL é compatível com certificados X.509, que são certificados digitais fornecidos e assinados por grandes entidades certificadoras no mundo.

Para este trabalho utilizaremos um certificado digital gerado por uma ferramenta chamada OPENSSL, que foi escrita em linguagem C, sendo de código aberto e amplamente suportada pelos sistemas operacionais disponíveis.

3.1 PHP

PHP é uma linguagem de programação web que permite criar sites dinâmicos, possibilitando uma interação com o usuário através de formulários, parâmetros, e links.

A diferença do PHP com relação a linguagens semelhantes como o JavaScritp é que o código PHP é executado no servidor, sendo enviado para o cliente apenas o código HTML. Desta maneira é possível interagir com o banco de dados existente no servidor, com a vantagem de não expor o código fonte da aplicação. Neste trabalho o PHP será responsável por fazer a autenticação do usuário e interagir com o *iptables*.

3.3 FIREWALL

Firewall é um dos principais mecanismos utilizados para minimizar o risco de ataques bem sucedidos. Em sua implementação padrão ele esta situado entre duas redes, sendo o agente que irá permitir ou não o acesso, fazendo uma analogia ao seu nome ele atua como uma parede corta-fogo entre duas redes. Podendo ser divididos em dois tipos: dispositivos de hardware e dispositivos de software.

Firewall de softwares: são programas de computador que podem ser instalados em um sistema operacional, por este motivo é necessário uma atenção adicional á segurança do próprio sistema operacional.

Firewall de hardware: são dispositivos mistos, eles integram o software com o hardware de forma que os mesmos sejam compatíveis entre si e feitos um para outro. Em geral apresenta desempenho mais elevado.

Estes *firewalls* podem operar basicamente de duas maneiras, nível de aplicação e/ou nível de pacotes.

No nível de aplicação, estes *firewalls* são mais intrusivos, pois analisam o conteúdo do pacote para tomada de decisão e permitem um controle do conteúdo do tráfego. *Proxy* é um exemplo deste tipo de *firewall*.

O *iptables* em sua implementação tradicional, que é o *firewall* mais utilizado junto com o Linux, trabalha no nível dos pacotes.

Um pacote é composto de duas partes, o *Header* (cabeçalho), e o *Body* (corpo). O que o *firewall* analisa são os cabeçalhos de cada pacote trafegado, verificando origem/porta, destino/porta, estado da conexão entre outros. Um exemplo é quando vamos fazer um download de um arquivo ou qualquer outra informação trafegada na rede, o que realmente está acontecendo é que o arquivo é dividido em pacotes, ou seja, fragmentos de 1500 bytes (MTU da rede). O *iptables* opera na camada de rede e transporte do modelo *Open Systems Interconnection* (OSI).

Podemos configurar a política básica das regras do *firewall* de duas maneiras. A primeira mais usual e muito mais segura, que é bloquear todo tráfego e liberar somente o necessário. A segunda não tão usual, mais válida que consiste em liberar tudo, bloqueando apenas o que não interessa.

Outra habilidade que o *iptables* possui é a possibilidade de criar logs de eventos do sistema. Desta maneira temos um registro das tentativas realizadas, das decisões do *firewall*, e de sua eficiência.

O *iptables* é uma ferramenta baseada em linha de comando, é necessário estar utilizando o sistema Linux com permissão de super usuário (root) para realizar alterações em suas tabelas ou regras.

O *iptables* é organizado por regras e cadeias, onde regras são instruções dadas ao *firewall*, e cadeias são locais onde as regras são processadas em ordem pelo *firewall*. Toda cadeia tem uma politica padrão, definida pelo usuário, a cadeia é percorrida até uma regra ser atingida, as demais são ignoradas.

Se nenhuma regra for atingida, usa-se a regra padrão.

O *iptables* organiza o fluxo de dados de pacotes em tabelas, que são cadeias pré-definidas, como demostrado na tabela abaixo:

Tabela 1 – Tabela FILTER.

Tabela	Descrição
FILTER	É a tabela padrão com três cadeias responsável por filtrar os pacotes; quando não especificada na montagem da regra, é ela que será utilizada.
Cadeias	Descrição
INPUT	Onde é analisado o tráfego que tem como destino a própria máquina do firewall.
OUTPUT	Verifica o tráfego gerado localmente, com destino local ou remoto.
FORWARD	Tráfego passante entre as redes ligadas no firewall.

Os elementos mais comuns para a tabela FILTER são explicados na Tabela 2:

Tabela 2 – Opções tabela FILTER.

-s	(SOURCE) Origem, estabelece a origem do pacote, geralmente é o endereço IP.	-s 10.1.1.1 -s 10.1.1.1/255.255.255.255 10.1.1.0/255.255.255.0
-d	(DESTINATION) Estabelece o destino do pacote.	Sintaxe igual ao –s
-p	(PROTOCOL) Especifica o protocolo a ser filtrado.	-p icmp -p tcp -p udp
-i	(IN-INTERFACE) Especifica a interface de entrada.	-i eth0
-o	(OUT-INTERFACE) Especifica a	-i eth1

	interface de saída.	
!	Exclusão, utilizado com –s, -d, -p, -i, para excluir o argumento.	-s ! 10.0.0.1 #Qualquer entrada exceto 10.0.0.1
--ssport	(SOURCE PORT), Porta de origem, Deve ser utilizada com as opções –p udp, -p tcp	-p tcp –sport 80
--dport	(DESTINATION PORT), Porta de destino, deve ser utilizada com as opções –p udp, -p tcp	Sintaxe igual ao –ssport

Tabela 3 – Tabela NAT.

Tabela	Descrição
NAT	Altera a origem e destino do pacote. Exemplo: é utilizada na passagem de informações entre duas redes (rede local e internet).
Cadeias	Descrição
PREROUTING	Usado para alterar o destino do pacote, analisando pacotes que estão entrando no *kernel* para sofrer NAT. (DNAT)
POSTROUTING	Usado para alterar o endereço de origem, analisando pacotes que estão saindo do *kernel* após sofrerem NAT.(SNAT)
OUTPUT	Qualquer pacote gerado localmente. (DNAT)

Os elementos mais comuns para esta tabela são:

Tabela 4 – Opções tabela NAT.

--t	(TABLE), especifica a tabela a ser utilizada, a tabela padrão é a INPUT	#iptables –t nat –A
--to	Utilizado para definir IP e porta de destino após um PREROUTING, ou de origem após um POSTROUTING ou OUTPUT	-j DNAT –to 10.1.1.5 -j DNAT –to 10.1.1.15:8080 -j SNAT –to 10.1.1.4
--dport	Define a porta de destino	

Para ativar o NAT é necessário carregar um modulo especifico, para isso devemos utilizar o comando:

#modprobe iptable_nat

Existe uma tabela chamada *RAW*, que é utilizada para marcar os pacotes para analise posterior, e a tabela *MANGLE*, que altera as informações do pacote geralmente sendo implementadas junto com regras de QoS. Estas tabelas não serão discutidas, pois não serão utilizadas para a implementação deste trabalho.

Segundo ESQUIVEL(2006), o funcionamento básico do *iptables* pode ser resumido da seguinte forma:

- Quando um pacote é recebido pela placa de rede, o *kernel* primeiramente verifica qual é o seu destino;
- Se o destino for o próprio equipamento, o pacote é passado para a cadeia INPUT. Se ele passar pelas regras dessa cadeia, ele será repassado para o processo de destino local, que está esperando pelo pacote.
- Se o *kernel* não tiver o *forwarding* habilitado ou se não souber como encaminhar esse pacote, este será descartado. Se o *forwarding* estiver habilitado para outra interface de rede, o pacote irá para a cadeira FORWARD. Se o pacote passar pelas regras dessa cadeia, ele será aceito e repassado adiante. Normalmente, essa é a cadeia utilizada quando o Linux funciona como um *firewall*.
- Um programa sendo executado no equipamento pode enviar pacotes à rede, que são enviados à cadeia OUTPUT. Se esses pacotes forem aceitos pelas regras existentes nessa cadeia, serão enviados por meio da interface.

A ação deve ser aplicada através do parâmetro *"-j target*", onde substituímos *target* pela ação que estão descritas abaixo conforme FAVERO (2007):

ACCEPT: o pacote é aceito;

DROP: o pacote é descartado, não avisando a origem sobre o ocorrido;

REJECT: o pacote é descartado, porém um pacote ICMP com a mensagem de erro é enviado como retorno ao cliente. O tipo ICMP default é *icmp-port-unreachable*, porém através da opção *–rejectwith*, os seguintes tipos podem ser utilizados: *icmp-net-unreachable, icmp-host-unreachable, icmp-port-unreachable, icmp-proto-unreachable, icmp-net-prohibited, icmp-host-prohibited ou icmp-admin-prohibited;*

QUEUE: passa o pacote para algum processo em espaço de usuário;

LOG: registra algumas informações do pacote;

RETURN: para de verificar as regras desta *chain* e retorna à verificação na regra posterior à da *chain* anterior.

As regras do IPTABLES são processadas em sua ordem, ou seja, sempre valerá a primeira em caso de impasse entre regras. Assim entre as regras:

#iptables -A FORWARD -p icmp -j DROP
#iptables -A FORWARD -p icmp -j ACCEPT
Valerá:
#iptables -A FORWARD -p icmp -j DROP

Já entre as regras:

#iptables -A FORWARD -p icmp -j ACCEPT
#iptables -A FORWARD -p icmp -j DROP
Valerá:
#iptables -A FORWARD -p icmp -j ACCEPT

Na tabela 5 estão descritos as ações utilizado pelo *iptables* para montar suas regras:

Tabela 5 – Comandos para as regras *iptables*.

-A	adiciona ou atualiza uma regra no fim;
-I	apenas adiciona uma nova regra no início;
-D	exclui uma regra específica;
-P	define a regra padrão;
-L	lista todas as regras armazenadas;
-F	exclui todas as regras armazenadas;
-R	substitui uma regra armazenada;
-C	efetua uma checagem das regras básicas;
-N	cria uma regra com nome específico;
-X	exclui uma regra com nome específico

Alguns exemplos de regras utilizados neste trabalho estão descritas detalhadamente na tabela 6.

Tabela 6 – Exemplos de Regra IPTABLES.

IPTABLES –A INPUT –p TCP --dport 80 -j ACCEPT # Permitir acesso a porta 80
IPTABLES –A INPUT –p TCP --dport 21 -j DROP #Nega acesso a porta 21

IPTABLES –A FOWARD –p TCP –dport 3389 –j ACCEPT # Permite o encaminhamento de pacotes para a porta 3389
iptables -P INPUT DROP *iptables -P FORWARD DROP* *iptables -P OUTPUT DROP* # Seta a regra padrão das tabelas INPUT, FORWARD,OUTPUT para DROP, ou seja, # qualquer trafego é negado por padrão.
iptables -t nat -A PREROUTING -p tcp -d 10.0.0.2 --dport 22 -j DNAT --to 192.168.0.3:22 #Redireciona todos os pacotes destinados à porta 22 da máquina 10.0.0.2 para a máquina #192.168.0.3. Esse tipo de regra exige a especificação do protocolo.
iptables -t nat –L iptables -t filter –L #Lista as regras da tabela NAT e FILTER

3.3.1 ROTEAMENTO DINAMICO

Quando existe mais de uma sub-rede envolvida no processo, é necessário que o roteamento seja ativado para que o *iptables* funcione corretamente, ele será responsável pelo encaminhamento de pacotes de uma rede para outra. O roteamento dinâmico, via *kernel*, pode ser ativado pelo comando:

#echo 1 > /proc/sys/net/ipv4/ip_forward

4. IMPLEMENTAÇÃO

Neste capítulo, será demonstrada a implementação do servidor Linux, utilizado como base uma distribuição do Debian. Nele foram instalados o servidor web Apache e desenvolvida a aplicação em PHP para interagir com o *iptables*.

4.1 CENARIO

Foi criado um cenário hipotético, com a seguinte estrutura:

- 1 (um) firewall com as redes 192.168.0.0/24 para sua rede interna, e com a rede 10.0.0.0/8 para simular sua rede WAN. Foi utilizado a distribuição Debian do sistema operacional Linux com duas placas de rede.

Servidor Terminal Server com o Windows 2008 Server com IP: 192.168.0.2 para simular o servidor de aplicação que deve ser protegido.

Estação cliente, que será utilizada para simular o usuário não autorizado enquanto ele ainda não estiver autenticado, e quando autenticar ele irá simular o usuário autorizado. Este cenário é exemplificado na figura 2, abaixo:

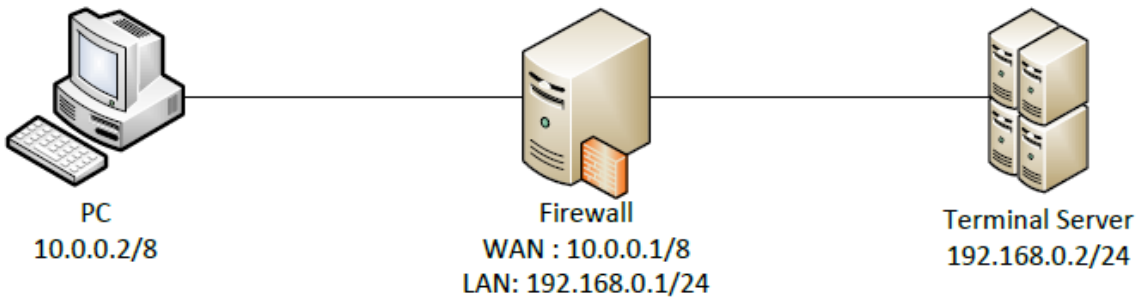

Figura 2 – Esboço do cenário proposto [Autoria própria].

4.2 ANALISE

Para a implementação do projeto foi realizado um trabalho de análise dos requisitos, e através dela foi gerado um diagrama de uso demostrado na figura 3.

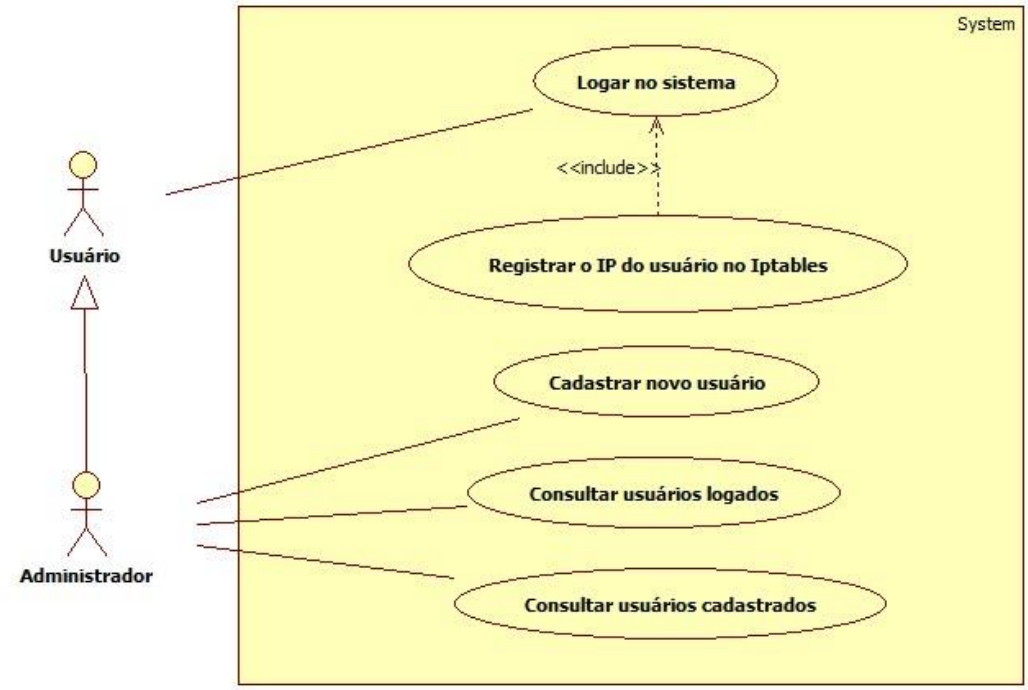

Figura 3 – Diagrama de caso de uso [Autoria própria].

Nesse levantamento resultou em uma sequência de passos onde o administrador e o usuário são *Actor* do sistema. O usuário é responsável por entrar com os dados (nome de usuário e senha),e o administrador quando já autenticado no sistema pode cadastrar novos usuários, visualizar os usuários cadastrados com suas respectivas portas, excluir ou alterar um usuário. O administrador conta também com um relatório dos últimos acessos do sistema.

Quando o usuário informa seus dados corretamente, o sistema captura o IP de sua conexão, consulta no bancos de dados qual a porta liberada para este usuário, e adiciona uma regra no *firewall iptables* permitindo que o IP do usuário acesse uma porta que esta fechada para outros IPs.

A análise realizada demostra na figura 4, o fluxograma do funcionamento do sistema. O usuário deve autenticar-se no sistema, informando seu nome usuário e senha. O sistema por sua vez consulta no banco de dados, se a informação for verdadeira ele registra seu IP no *iptables.* Caso a informação de usuário e senha estiverem incorretas é apresentada uma tela de erro.

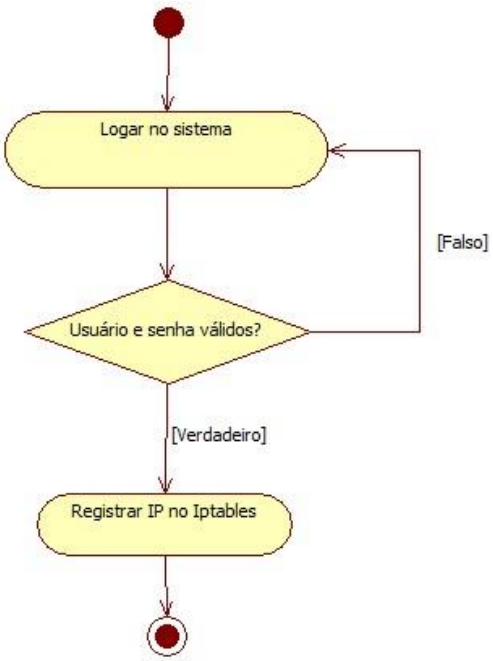

Figura 4 – Diagrama de Estado[Autoria própria].

4.3 IMPLEMENTAÇÃO

Os itens abaixo mostram e descrevem as telas do sistema.

4.3.1 TELAS DO SISTEMA

A primeira tela do sistema é a de autenticação (página inicial). Ela solicita que o usuário informe seu nome de usuário e sua senha. Após clicar no botão entrar, estas informações são validadas em uma base de dados, caso estejam corretas a figura 7 é exibida, caso incorretas a figura 6 é mostrada ao usuário. Abaixo a figura 5 que contêm a página inicial.

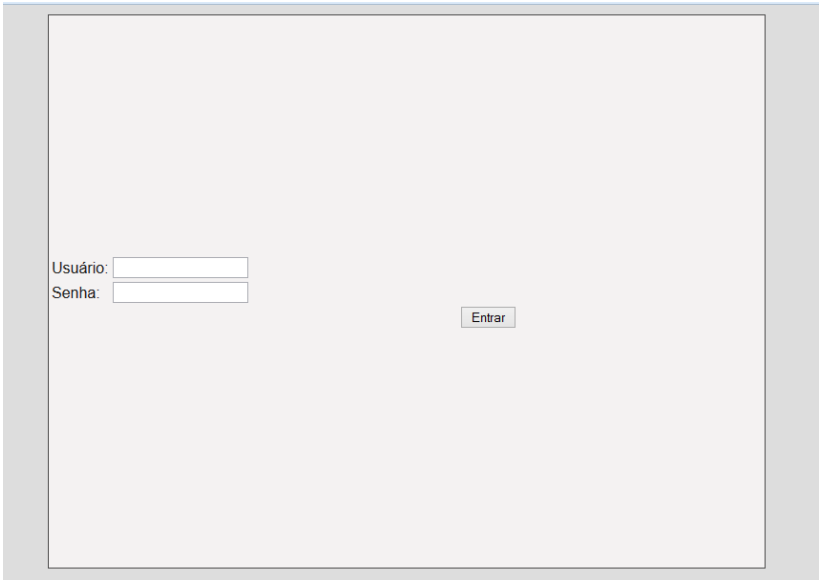

Figura 5 – Tela de Login e Senha do sistema[Autoria própria].

A figura 6 mostra a tela responsável por informar que o usuário entrou com usuário e/ou senha inválidos.

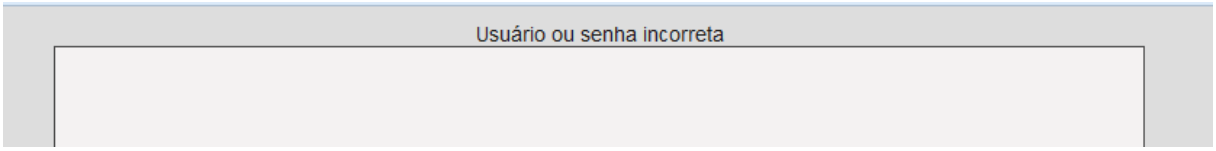

Figura 6 – Tela usuário e senha inválidos [Autoria própria].

Quando o usuário obtém sucesso na validação de seus dados a página mostrada é demostrada na figura 7. Nela são exibidas algumas informações:

- 1 – Nome do Usuário;
- 2 – Opção para sair do sistema;
- 3 – IP do cliente (usuário que fez a autenticação);
- 4 – Porta destinada ao acesso;
- 5 – Data e hora da autenticação;

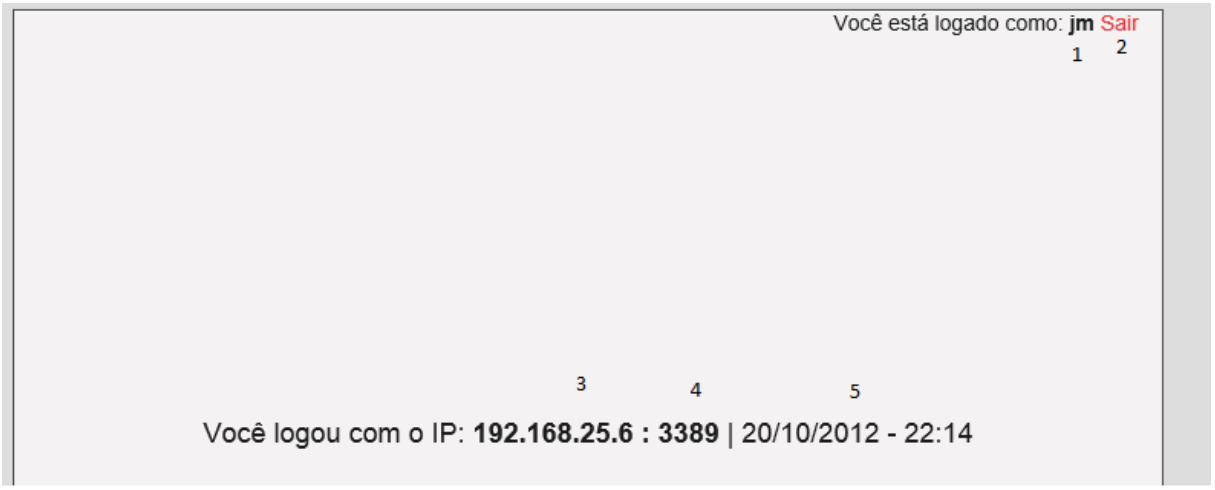

Figura 7 – Usuário Logado[Autoria própria].

A partir deste momento onde a tela da figura 7 é exibida, o IP 192.168.24.6, tem acesso ao NAT do firewall para a porta 3389. Quando autenticamos no sistema como usuário administrador, a tela inicial ganha uma nova opção: Gerenciar, como é exibido na figura 8.

Figura 8 – Admin logado [Autoria própria].

Quando o administrador clica na opção gerenciar, ele tem acesso á página de controle da aplicação. Nesta tela ele tem a opção para cadastrar um novo usuário, onde ele deve informar o nome do usuário, senha, e a porta que este usuário terá acesso. Estas informações são armazenas em um arquivo texto, sendo a senha de forma criptografada. Nesta mesma tela o administrador pode visualizar os usuários cadastrados, podendo excluir e verificar o relatório dos últimos usuários que entraram no sistema. A figura 9 mostra a tela de administração da aplicação.

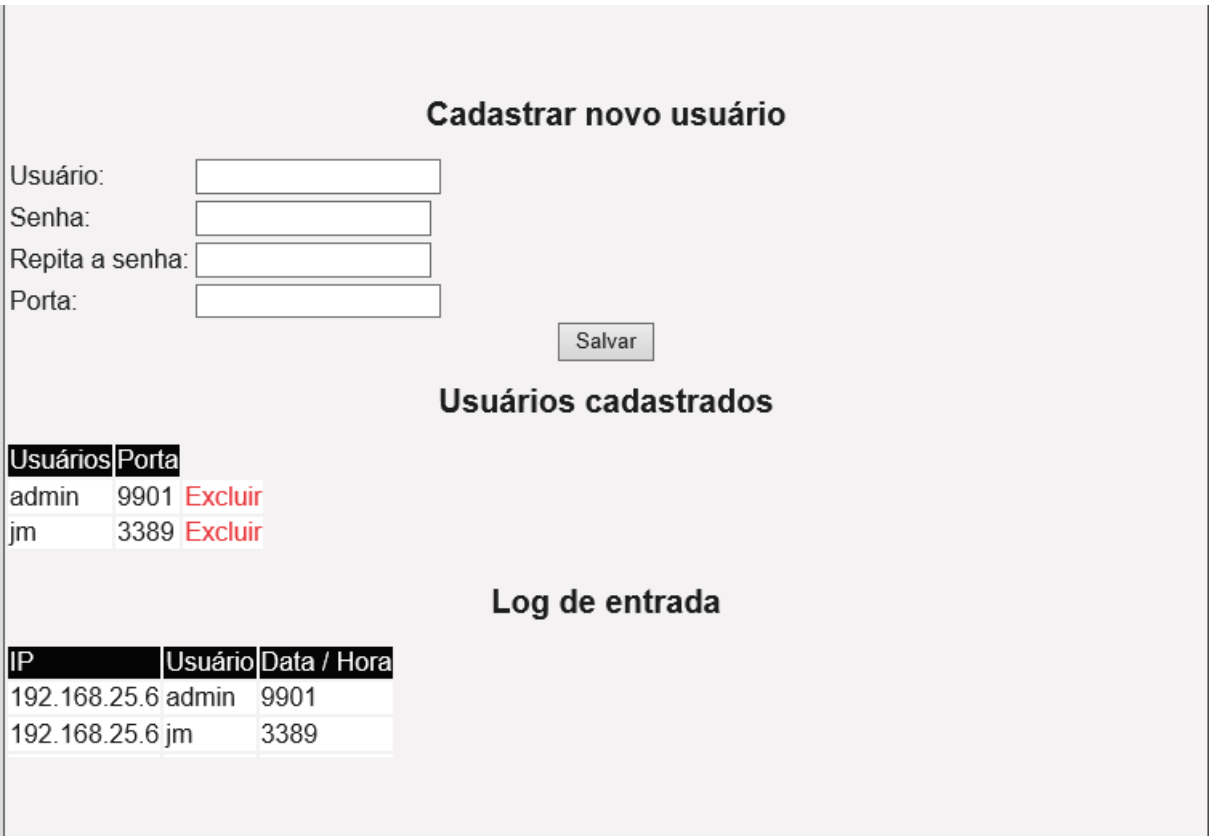

Figura 9 – Tela de gerenciamento [Autoria própria].

O cenário proposto busca proteger um servidor de terminal, para isso fizemos o teste utilizando um cliente não autenticado, que tenta acessar o IP 10.0.0.1, IP correspondente a placa de rede da WAN do *firewall*. Na figura 10 mostramos que ele não conseguiu acesso pois o IP do cliente ainda não esta liberado no *firewall*, uma vez que ele ainda não autenticou.

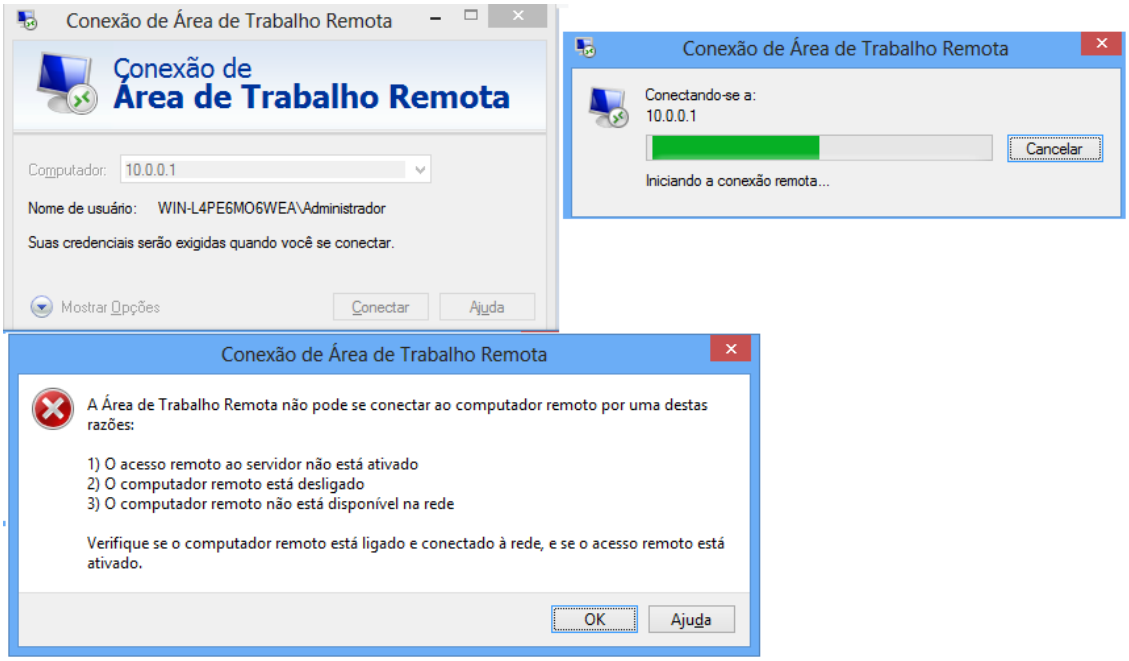

Figura 10 – Acesso negado [Autoria própria].

Quando um usuário que tenha a porta 3389 cadastrada autenticar, seu IP será adicionado no *firewall*, a partir deste momento o acesso é permitido, como mostra a figura 11, onde é solicitado pelo servidor de terminal, o usuário e senha Windows necessário para acesso ao servidor.

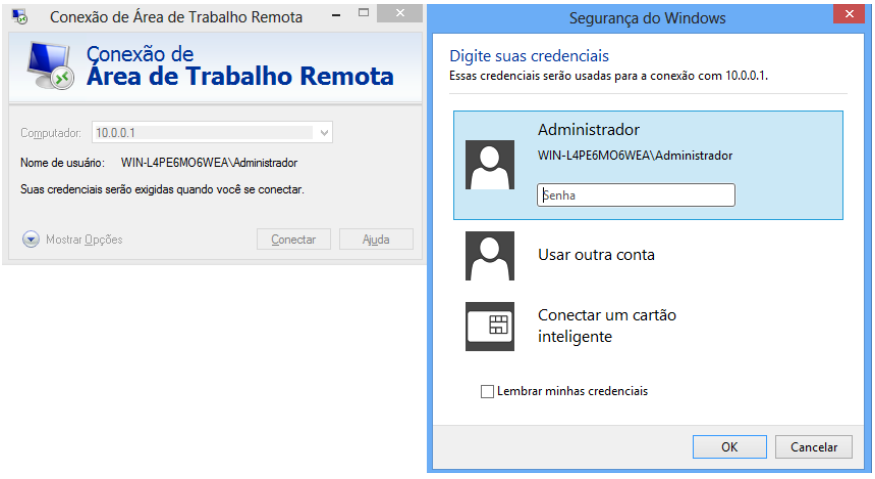

Figura 11 – Acesso permitido [Autoria própria].

4.3.2 FUNCIONAMENTO INTERNO

Este capítulo explica detalhadamente o funcionamento da aplicação, mostrando como ela interage com o *iptables*, e outros componentes Linux necessários.

4.3.2.1 CRIPTOGRAFIA

Para que os dados de *login* e senha trafeguem pela página de forma segura, impossibilitando que sejam recuperadas por um *sniffer* ou outro dispositivo inserido entre a comunicação, foi ativada a criptografia do Apache utilizando um certificado gerado pelo OpenSSL. Primeiramente devemos gerar o certificado, utilizando o seguinte comando:

#openssl req $@ -new -x509 -days 365 -nodes -out /etc/apache2/apache.pem -keyout /etc/apache2/apache.pem

Para gerar o certificado é necessário informar os dados do certificado, como: país, estado, cidade, nome da empresa, unidade da organização, nome do usuário, e-mail. Após estes dados informados o certificado é gerado na pasta /etc/apache2/apache.pem. A figura 12 demostra a criação do certificado utilizado pelo apache para criptografar a comunicação ente cliente e servidor web.

```
root@debian:/etc/apache2# openssl req $@ -new -x509 -days 365 -nodes -out /etc/apache2/apache.pem -keyout /etc/apache2/apache.pem
Generating a 1024 bit RSA private key
..........++++++
.............................++++++
writing new private key to '/etc/apache2/apache.pem'
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:BR
State or Province Name (full name) [Some-State]:PR
Locality Name (eg, city) []:Ponta Grossa
Organization Name (eg, company) [Internet Widgits Pty Ltd]:Teste
Organizational Unit Name (eg, section) []:Teste
Common Name (eg, YOUR name) []:Teste
Email Address []:Teste@teste.com
root@debian:/etc/apache2#
```

Figura 12 – Criação do certificado [Autoria própria].

É necessário habilitar o Apache para suportar a criptografia utilizando o certificado SSL gerado. Devemos carregar módulo demonstrado na figura 13.

#a2enmod ssl

```
root@debian:/etc/apache2# a2enmod ssl
Module ssl already enabled
root@debian:/etc/apache2#
```

Figura 13 – Modulo a2enmod ssl para o apache [Autoria própria].

Devemos modificar o arquivo do site padrão para apontar o certificado SSL, para isso temos que editar /etc/apache/sites-available/default:

Em <VirtualHost *:80> mudar para <VirtualHost *:443>

e adicionar as seguintes linhas:

SSLEngine on

ServerSignature On

SSLCertificateFIle /etc/apache2/apache.pem

No arquivo /etc/apache/ports.conf para ele escutar a porta 443

Em NameVirtualHost *:80 mudar para NameVirtualHost *:443

Agora basta reiniciar o apache com o comando:

Apache2ctl restart

4.3.2.2 CONFIGURAÇÃO DA REDE

Devemos editar o arquivo /etc/networks/interface para configurar as placas de rede: Para isso devemos configurar os parâmetros para ETH0 e EHT1, como exemplificado na figura 14.

A Sintaxe do comando é

Iface (placa de rede) inet static

adress (endereço ip)

netmask (mascara de rede)

```
#s file describes the network interfaces available on your system
# and how to activate them. For more information, see interfaces(5).

# The loopback network interface
auto lo
iface lo inet loopback

# The primary network interface
#allow-hotplug eth0
#iface eth0 inet dhcp

auto eth0
iface eth0 inet static
        address 10.0.0.1
        netmask 255.0.0.0

auto eth1
iface eth1 inet static
        address 192.168.0.1
        netmask 255.255.255.0
~
~
~
"/etc/network/interfaces" 20 lines, 426 characters
```

Figura 14 – Configuração das interfaces [Autoria própria].

4.3.2.3 PERMITIR APACHE INTERAGIR COM IPTABLES

Para permitir que o que o apache possa alterar as configurações de *firewall*, temos que permitir que seu usuário altere o *iptables*, que por padrão só permite que o super usuário tenha controle sobre ele.

O *software* que controla isso no Linux é o SUDO, que na distribuição do Debian 6 não é um software padrão apesar de ser amplamente utilizado. Para instalar o SUDO devemos utilizar o seguinte comando, como mostra a figura 15.

Apt-get install sudo

```
root@debian:~# apt-get install sudo
Lendo listas de pacotes... Pronto
Construindo Ã¡rvore de dependÃªncias
Lendo informaÃ§Ã£o de estado... Pronto
Os NOVOS pacotes a seguir serÃ£o instalados:
  sudo
0 pacotes atualizados, 1 pacotes novos instalados, 0 a serem removidos e 1 nÃ£o
atualizados.
Ã preciso baixar 610 kB de arquivos.
Depois desta operaÃ§Ã£o, 983 kB adicionais de espaÃ§o em disco serÃ£o usados.
Obter:1 http://security.debian.org/ squeeze/updates/main sudo amd64 1.7.4p4-2.sq
ueeze.3 [610 kB]
Baixados 610 kB em 2s (222 kB/s)
Selecionando pacote previamente nÃ£o selecionado sudo.
(Lendo banco de dados ... 57478 ficheiros e directÃ³rios actualmente instalados.
)
Desempacotando sudo (de .../sudo_1.7.4p4-2.squeeze.3_amd64.deb) ...
Processando gatilhos para man-db ...
Configurando sudo (1.7.4p4-2.squeeze.3) ...
No /etc/sudoers found... creating one for you.
root@debian:~#
```

Figura 15 – Instalação do SUDO [Autoria própria].

Após o *sudo* instalado devemos acessar o software *VISUDO*, que é um editor de texto para o arquivo /etc/sudores. É neste arquivo que é configurado que usuários não root podem fazer determinada ação. Neste caso configuraremos para que o usuário do Apache possa interagir com o *iptables*. Devendo ser adicionado a seguinte linha:

user ALL=(ALL) NOPASSWD: /sbin/iptables, /sbin/iptables-save, /sbin/iptables-restore

4.3.2.4 SCRIPTS PARA IPTABLES

4.3.2.4 PAGINAS PHP

As páginas desenvolvidas em PHPs ficam localizadas em /var/www/, diretório padrão do apache2. Nela existem 3 arquivos principais como descrito na tabela 7:

Tabela 7 – Páginas PHP

Index.php	Arquivo inicial, solicita usuário e senha, depois de informado valida se o usuário e senha estão corretos, redireciona para a página home.php, caso contrario ele exibe usuário/senha inválido.
Home.php	Esta página mostra que o usuário esta logado, é responsável por editar o arquivo texto /var/www/arquivos/users_ip que adiciona o IP real do usuário em uma lista. Também aciona o script /var/www/arquivos/firewallscript que é responsável por adicionar o IP real do usuário ao *firewall iptables*.
Users.php	Página de administração, que permite adicionar/remover os usuários, e exibe os usuários logados. Apenas o usuário Admin tem acesso.

Na tabela 8 são descritos os scripts e arquivos auxiliares:

Tabela 8 – arquivos auxiliares

firewall	Este arquivo contêm a configuração do firewall geral, ele deve ser criado pelo administrador do servidor, utilizando as regras básicas do *iptables*. Nele deve ser feito as opções de NAT, a página irá apenas adicionar os IPs reais que podem acessar a regra de NAT já definida.
firewallscript	Script responsável por ler o arquivo Users_ip, e adicionar no *iptables* os IPs verdadeiros(clientes) que podem ter acesso á regra de NAT já criada no arquivo firewall.
Users	Arquivo utilizado pela aplicação para armazenar os usuário e senha, e porta do NAT, tendo a seguinte sintaxe: Usuário : Senha Criptografada : Porta
Users_ip	Arquivo que mantêm os usuário logados, ele é utilizado pelo arquivo firewallscript para adicionar os IPS no Iptables, ele tem a seguinte sintaxe: IP Cliente : Usuário : Porta : Data/Hora Logon
Horafirewall.php	Responsável por analisar o arquivo Users_ip e remover os usuário autenticados a mais de 6 horas.

4.3.2.4 AGENDAMENTO DE SCRIPT

Quando um usuário é autenticado e seu IP registrado no *firewall*, ele fica valido por 6 horas, quem controla isso é a página horafirewall.php, o código desta pagina executa uma verificação do arquivo Users_ip, apagando usuários logados a mais de 6 horas. Ela é executada a cada 30 minutos de forma automatizada.

O Crontab utilitario do Linux para tarefas agendadas deve ser configurado, para isso devemos edita-lo utilizando o comando:

Crontab –e

Adicionando as seguintes linhas:

00 * * * * php /var/www/horafirewall.php

30 * * * * php /var/www/horafirewall.php

Estas linhas são responsáveis por executar a pagina horafirewall.php a cada meia hora.

5. CONCLUSÃO

A necessidade de segurança nos sistemas de informação é fator de muita preocupação para grandes organizações, o que não ocorre nas organizações de menor porto onde o investimento em segurança é mínimo. A obrigatoriedade de segurança se deve ao fato de segredos operacionais, e informações de alto valor serem armazenadas em formato digital. Este tema tende a crescer a cada dia considerando o fato que, hoje mesmo as pequenas organizações sem recursos significativos destinados á tecnologia da informação, necessitarem compartilhar sistemas de acesso remoto com colaboradores remotos, parceiros e fornecedores.

Através da implementação deste serviço de autenticação extra para acesso a uma porta em um servidor, é possível melhorar a segurança, não deixando portas abertas com recursos externos vulneráveis para toda a internet diretamente. A aplicação que atua em conjunto com a proteção tradicional do *firewall iptables* fornece uma maneira auxiliar para dificultar um invasor externo de conseguir êxito no acesso a informações da empresa que são compartilhadas através de sistemas da informação com colaboradores remotos, parceiros e fornecedores.

Este sistema é útil quando intencionamos melhorar a proteção contra ataques conscientes, onde o invasor busca uma informação específica e tem um objetivo maior que diversão ou estudo. Ataque inconsciente são usuários que atacam um servidor, sem saber qual é o mesmo, apenas por diversão ou estudo, deixando o servidor sem acesso, apagando informações ou disponibilizando os dados pela rede.

Como este trabalho utiliza tecnologias livres, não temos custo com licenças de *software*, e por isso esta proposta é atrativa para pequenas organizações. O Administrador de rede também fica em uma situação mais confortável, pois sabe que pessoas mal intencionadas terão maior dificuldade no acesso. Como as portas inicialmente encontram-se bloqueadas isso dificulta o trabalho de softwares que localizam portas abertas, sendo que elas só são liberadas para os IP dos clientes que realmente precisam fazer o acesso em certo tempo. O administrador de rede, pode utilizar as informações geradas por esta ferramenta para ter um log de acesso aos recursos internos de sua rede, e assim fornecer informações mais precisas.

Este trabalho pode ser evoluído, para uma plataforma que forneça criptografia variável em cima de uma transação TCP/IP transparente para o usuário, sendo assim mais uma opção para substituir a VPN.

REFERÊNCIAS

ALBUQUERQUE, Ricardo e RIBEIRO, Bruno. **SEGURANÇA NO DESENVOLVIMENTO DE SOFTWARE** – Como desenvolver sistemas seguros e avaliar a segurança de aplicações desenvolvidas com base na ISO 15.408. Editora Campus. Rio de Janeiro, 2002.

CARUSO, Carlos A. A., STEFFEN, Flavio D. **SEGURANÇA EM INFORMÁTICA E DE INFORMAÇÕES.** São Paulo: SENAC, 1999.

DIAS, Cláudia. **SEGURANÇA E AUDITORIA DA TECNOLOGIA DA NFORMAÇÃO**. Axcel Books. Rio de Janeiro, 2000.

ESQUIVEL, Claudyson Jonathas. **GERENCIAMENTO DE REGRAS DE FIREWALL IPTABLES EM AMBIENTE LINUX,** Uberlândia, 2006.

FÁVERO, André Luís **METODOLOGIA PARA DETECÇÃO DE INCOERÊNCIAS ENTRE REGRAS EM FILTROS DE PACOTES**, Dissertação apresentada como requisito parcial para a obtenção do grau de Mestre em Ciência da Computação, Porto Alegre, 2007.

FERNADES, Jocimar. **CRIPTOGRAFIA E O MODELO CRIPTOGRÁFICO DO SISTEMA INFORMATIZADO DE ELEIÇÕES DO BRASIL.** 2007. 59p. Trabalho de conclusão de curso(Graduação Lato Sensu em Redes de Computadores) – ESAB – Escola Superior Aberta do Brasil, Vitoria, 2007.

KRAUSE, Micki e TIPTON, Harold F. **HANDBOOK OF INFORMATION SECURITY MANAGEMENT**. Auerbach Publications, 1999.

LAUREANO, Marcos Aurelio Pchek e MORAES, Paulo Eduardo Sobreira. **SEGURANÇA COMO ESTRATÉGIA DE GESTÃO DA INFORMAÇÃO**, Revista Economia & Tecnologia – ISSN 1415-451X Vol. 8 – Fascículo 3 – P. 38-44,2005.

LAUREANO, Marcos Aurelio Pchek. **FIREWALL COM IPTABLES NO LINUX**,2002.

MACHADO, Leandro Alves. **DESENVOLVIMENTO DE UMA INTERFACE WEB PARA A APRESENTAÇÃO ESPACIAL DE DADOS DAS PLATAFORMAS DE COLETA DE DADOS AMBIENTAIS.** 2010. Monografia apresentada ao curso de Ciência da Computação da Faculdade Lourenço Filho, Fortaleza, 2010.

NAKAMURA, Emilio T.; GEUS, Paulo L. **SEGURANÇA DE REDE EM AMBIENTES COOPERATIVOS**. Novatec, São Paulo:, 2007.

NBR ISO/IEC 17799 – TECNOLOGIA DA INFORMAÇÃO. CÓDIGO DE PRÁTICA PARA GESTÃO DA SEGURANÇA DA INFORMAÇÃO. Associação Brasileira de Normas Técnicas. Rio de Janeiro, 2003.

REZENDE, Denis Alcides e ABREU, Aline França. **TECNOLOGIA DA INFORMAÇÃO APLICADA A SISTEMAS DE INFORMAÇÃO EMPRESARIAIS**. Editora Atlas. São Paulo,2000.

SÊMOLA, Marcos. **GESTÃO DA SEGURANÇA DA INFORMAÇÃO – UMA VISÃO EXECUTIVA**. Editora Campus. Rio de Janeiro, 2003.

SOARES, L. F. G.; LEMOS, G.; COLCHER, S. – **REDES DE COMPUTADORES DAS LANS, MANS E WANS ÀS REDES ATM**. 6º ed Rio de Janeiro: Campus, 1995. 576p.

SOUZA, Marcelo Soares**. ANÁLISE DO SERVIDOR WEB APACHE EM CLUSTERS OPENMOSIX COM MEMÓRIA COMPARTILHADA DISTRIBUÍDA.** CEBACAD Centro Baiano de Computação de Alto Desempenho Universidade Católica do Salvador (UCSal) Salvador.

WADLOW, Thomas. 2000. **Segurança de Redes.** Editora Campus. Rio de Janeiro.

ANEXOS

Pagina /var/www/home.php

```
<?php
if($_REQUEST['command'] == 'logout'){

	session_start();
	//sessÃ£o de usuÃ¡rio recebe Nulo
	 $_SESSION['user'] = NULL;
	//unset da sessÃ£o
	unset($_SESSION['user']);
	//redireciona para index
	session_destroy();
	echo "<script>location.href='./index.php'</script>";

}
	//inicia sessÃ£o
	session_start();
	//Se Session nÃ£o tiver nada em user, volta a pagina inicial
	if(!isset($_SESSION['user']))
	{
		session_destroy();
		echo "<script>location.href='./index.php'</script>";
		return;
	}
	//pega o ip da mÃ¡quina
	$ip = $_SERVER['REMOTE_ADDR'];
	//adiciona a url do arquivo users_ip a variÃ¡vel arquivo
	$arquivo = "/var/www/arquivos/users_ip";
	//abre o arquivo para leitura e joga no ponteiro
	$ponteiro = fopen($arquivo,"r");
	//variavel texto recebe o conteÃºdo do arquivo
	$texto = fread($ponteiro, filesize($arquivo));
	//fecha ponteiro
	fclose($ponteiro);
	//separa o conteÃºdo do arquivo por linhas
	$linha = explode("\n", $texto);
	//variÃ¡vel sinalizadora recebe 0
	$flag = 0;
	//variÃ¡vel user recebe valor da sessÃ£o
	$user = ' : '.$_SESSION['user']. ' : ';
	//monta o conteÃºdo que serÃ¡ gravado no arquivo / IP : UsuÃ¡rio : Data

//carrega porta

//arquivo recebe a url do arquivo users
$arquivousers = "/var/www/arquivos/users";
//abre o arquivo para leitura e joga no ponteiro
$ponteirousers = fopen($arquivousers,"r");
//variavel texto recebe o conteÃºdo do arquivo
```

```
$textousers = fread($ponteirousers, filesize($arquivousers));
//fecha ponteiro
fclose($ponteirousers);
//separa o conteÃºdo do arquivo por linhas
$linhausers = explode("\n", $textousers);
//percorre o vetor
for($i=0;$i<(sizeOf($linhausers));$i++)
{
//verifica se a linha contÃ©m o usuÃ¡rio que logou
if(strstr($linhausers[$i], $_SESSION['user'] , false))
	{
	//separa a linha por :
	$usuarioport = explode(" : ", $linhausers[$i]);
	}
}

	$conteudo = $ip." : ".$_SESSION['user']." : ". $usuarioport[2]. " : ".date('d/m/Y -
H:i')."\n";
	//percorre o vetor de linhas
	for($i=0;$i<(sizeOf($linha)-1);$i++){
	//se a linha em questÃ£o conter o valor da variÃ¡vel user
		if(strstr($linha[$i], $user, false)){
			//sinalizadora recebe 1
			$flag = 1;
			//dado recebe o conteÃºdo
			$dado .= $conteudo;
			//senao
		}else{
			//dado recebe a linha em questÃ£o
			$dado .= $linha[$i]."\n";
		}
	}
	//se sinalizadora for igual a 0
	if($flag == 0){
	//abre para escrita colocando o ponteiro no final do arquivo
	$ponteiro = fopen($arquivo,"a");
	//escreve o conteudo no ponteiro
	fwrite($ponteiro, $conteudo);
	//fecha o ponteiro
	fclose($ponteiro);
	//se sinalizadora for igual a 1
	}else{
	//abre para escrita apagando todo o conteÃºdo do arquivo
	$ponteiro = fopen($arquivo,"w+");
	//escreve dado no ponteiro
	fwrite($ponteiro, $dado);
	//fecha ponteiro
	fclose($ponteiro);
	}
```

```
echo "<html>"."\n";
echo "<head>"."\n";
echo '<meta http-equiv="Content-Type" content="text/html;charset=utf-8" >'."\n";
echo '<link rel="stylesheet" type="text/css" href="resources/style.css"/>'."\n";
echo "<title>AutenticaÃ§Ã£o de TS</title>"."\n";
echo "</head>"."\n";
echo "<body>"."\n";
echo '<div id="box">' . "\n";

//arquivo recebe a url do arquivo users
$arquivo = "/var/www/arquivos/users";
//abre o arquivo para leitura e joga no ponteiro
$ponteiro = fopen($arquivo,"r");
//variavel texto recebe o conteÃºdo do arquivo
$texto = fread($ponteiro, filesize($arquivo));
//fecha ponteiro
fclose($ponteiro);
//separa o conteÃºdo do arquivo por linhas
$linha = explode("\n", $texto);
//percorre o vetor
for($i=0;$i<(sizeOf($linha));$i++){
//verifica se a linha contÃ©m o usuÃ¡rio que logou
if(strstr($linha[$i], $_SESSION['user'] , false)){
//separa a linha por :
$usuario = explode(" : ", $linha[$i]);
}
}
```

Pagina /var/www/Index.php

```
<?php
//Se ja existir uma sessÃ£o aberta, redireciona para a pagina home
session_start();
if(isset($_SESSION['user']))
	echo "<script>location.href='./home.php'</script>";

if(isset($_POST['logon'])){
	//pega a url do arquivo users
	$arquivo = "/var/www/arquivos/users";
	//abre o arquivo para leitura e joga no ponteiro
	$ponteiro = fopen($arquivo,"r");
	//lendo o arquivo para a variÃ¡vel texto
	$texto = fread($ponteiro, filesize($arquivo));
	//fecha ponteiro
	fclose($ponteiro);
	//separa o conteÃºdo do arquivo por linhas
	$linha = explode("\n", $texto);
	//percorre o vetor de linhas do arquivo
	for($i=0;$i<(sizeOf($linha)-1);$i++){
```

```
        //separa o conteÃºdo da linha por ' : ' usuario : senha : porta
        $usuario = explode(" : ",$linha[$i]);
        //verifica se o usuÃ¡rio Ã© o mesmo que foi digitado na tela inicial
                if($usuario[0] == $_POST['user']){
                        //verifica se a senha Ã© igual a do usuÃ¡rio que foi digitado
                        if($usuario[1] == (md5($_POST['passwd']))){
                        //inicia sessao
                        session_start();
                        //sessao recebe usuÃ¡rio
                        $_SESSION['user'] = $_POST['user'];
                        //redireciona para home.php
                        echo "<script>location.href='./home.php'</script>";
                        }
                }else{
                        //sinalizadora recebe 0
                        $flag = 0;
                }
        }
        //se sinalizadora for 0
        if($flag == 0){
                //Mostra mensagem de usuÃ¡rio ou senha incorretos.
                echo "UsuÃ¡rio ou senha incorreta";
        }
}

//form de login
echo "<html>"."\n";
echo "<head>"."\n";
echo '<meta http-equiv="Content-Type" content="text/html;charset=utf-8" >'."\n";
echo '<link rel="stylesheet" type="text/css" href="resources/style.css"/>'."\n";
echo "<title>AutenticaÃ§Ã£o de TS</title>"."\n";
echo "</head>"."\n";
echo "<body>"."\n";
echo '<div id="box">' . "\n";
echo '<div id="login">';
echo "<form action='./' method='POST'>";
echo "<table id='table-login'>";
echo "<tr>";
echo "<td>UsuÃ¡rio: </td><td><input type='text' name='user' id='user'><td>";
echo "</tr>";
echo "<tr>";
echo "<td>Senha: </td><td><input type='password' name='passwd' id='passwd'></td>";
echo "</tr>";
echo "</table>";
echo "<input type='submit' name='logon' id='logon' value='Entrar'>";
echo "</form>";
echo "</div>";
echo "</div>";
echo "</body>"."\n";
echo "</html>"."\n";
```

```
?>
```

Pagina /var/www/users.php

```
<?php
	session_start();
	//Se Session nÃ£o tiver nada em user, volta a pagina inicial
	if(($_SESSION['user'])!="admin")
	{
		session_destroy();
		echo "<script>location.href='./index.php'</script>";
	}
//Salvar usuÃ¡rio
if(isset($_POST['save_user'])){
	//montando a linha de conteudo UsuÃ¡rio : Senha : IP : Porta
	$conteudo = $_POST['user']." : ". md5($_POST['passwd']). " : ".$_POST['port']."\n";
	//pega a url do arquivo users
	$arquivo = "/var/www/arquivos/users";
	//abre o arquivo para escrita e joga no ponteiro
	$ponteiro = fopen($arquivo,"a");
	//escreve o conteudo no ponteiro
	fwrite($ponteiro, $conteudo);
	//fecha ponteiro
	fclose($ponteiro);
}
//RemoÃ§Ã£o do UsuÃ¡rio
else if($_REQUEST['op'] == 'remove'){
	//pega a url do arquivo users
	$arquivo = "/var/www/arquivos/users";
	//abre o arquivo para leitura e joga no ponteiro
	$ponteiro = fopen($arquivo,"r");
	//lendo o arquivo para a variÃ¡vel texto
	$texto = fread($ponteiro, filesize($arquivo));
	//fecha ponteiro
	fclose($ponteiro);
	//separa o conteÃºdo do arquivo por linhas
	$linha = explode("\n", $texto);
	//variÃ¡vel usuÃ¡rio recebe por REQUEST o usuÃ¡rio que Ã© para ser excluÃ-do
	$usuario = $_REQUEST['user']." : ";
	//percorre o vetor de linhas do arquivo
	for($i=0;$i<(sizeOf($linha)-1);$i++){
		//se o conteÃºdo da variÃ¡vel usuario estiver em alguma das posiÃ§Ãµes do vetor
de linhas
		if(strstr($linha[$i], $usuario, false)){
		// content que serÃ¡ gravado no arquivo users recebe vazio
			$content .= '';
		//senao
		}else{
		 //content recebe a posiÃ§Ã£o do vetor de linha
			$content .= $linha[$i]."\n";
```

```
            }

        }
        //abre o arquivo para escrita e joga no ponteiro
        $ponteiro = fopen($arquivo,"w+");
        //grava a variÃ¡vel content no ponteiro de arquivo
        fwrite($ponteiro, $content);
        // fecha ponteiro
        fclose($ponteiro);
        //redireciona para pÃ¡gina users.php
        echo "<script>location.href='./users.php'</script>";
}

// Form de AdiciÃ§Ã£o de novo usuÃ¡rio
echo "<html>"."\n";
echo "<head>"."\n";
echo '<meta http-equiv="Content-Type" content="text/html;charset=utf-8" >'."\n";
echo '<link rel="stylesheet" type="text/css" href="resources/style.css"/>'."\n";
echo "<title>AutenticaÃ§Ã£o de TS</title>"."\n";
echo "</head>"."\n";
echo "<body>"."\n";
echo '<div id="box">' . "\n";
echo '<div id="new_user">';
echo "<h1>Cadastrar novo usuÃ¡rio</h1>";
echo "<form action='./users.php' method='POST'>";
echo "<table>";
echo "<tr>";
echo "<td>UsuÃ¡rio: </td><td><input type='text' name='user' id='user'><td>";
echo "</tr>";
echo "<tr>";
echo "<td>Senha: </td><td><input type='password' name='passwd' id='passwd'></td>";
echo "</tr>";
echo "<tr>";
echo "<td>Repita a senha: </td><td><input type='password' name='passwd2'
id='passwd2'></td>";
echo "</tr>";
echo "<tr>";
echo "<td>Porta: </td><td><input type='text' name='port' id='port'></td>";
echo "</tr>";
echo "</table>";
echo "<input type='submit' name='save_user' id='save_user' value='Salvar'>";
echo "</form>";
echo "</div>";

//pega a url do arquivo users
$arquivo = "/var/www/arquivos/users";
//abre o arquivo para leitura e joga no ponteiro
$ponteiro = fopen($arquivo,"r");
//adiciona a variÃ¡vel texto o conteÃºdo do arquivo
$texto = fread($ponteiro, filesize($arquivo));
```

```
//fecha ponteiro
fclose($ponteiro);
//separa o conteÃºdo do arquivo por linhas
$linha = explode("\n", $texto);

//Tabela de usuÃ¡rios
echo "<h1>UsuÃ¡rios cadastrados</h1>";
echo "<table id='usuarios' cellspacing='2'>";
//CabeÃ§alho
echo "<tr class='cab'><td>UsuÃ¡rios</td><td>Porta</td></tr>";
//percorre o vetor de linhas
for($i=0;$i<(sizeOf($linha)-1);$i++){
	//separa o conteÃºdo da linha por ' : ' usuario : senha : ip : porta
	$user = explode(" : ",$linha[$i]);
	//Resultado: UsuÃ¡rio Porta Excluir
	echo "<tr><td>".$user[0]."</td><td>".$user[2]."</td><td><a
href='./users.php?op=remove&user=".$user[0]."'>Excluir</a></td></tr>";
}
//Fim da tabela
echo "</table>";

//Tabela mostrando os usuÃ¡rios e os Ips
echo "<div id='logs'>";
echo "<h1>Log de entrada</h1>";
echo "<table id='user_ip' cellspacing='2'>";
//CabeÃ§alho
echo "<tr class='cab'><td>IP</td><td>UsuÃ¡rio</td><td>Data / Hora</td></tr>";
//arquivo recebe a url do arquivo users_ip
$arquivo = "/var/www/arquivos/users_ip";
//abre o arquivo para leitura e joga no ponteiro
$ponteiro = fopen($arquivo,"r");
//variavel texto recebe o conteÃºdo do arquivo
$texto = fread($ponteiro, filesize($arquivo));
//fecha ponteiro
fclose($ponteiro);
//separa o conteÃºdo do arquivo por linhas
$linha = explode("\n", $texto);
//percorre o vetor
for($i=0;$i<(sizeOf($linha));$i++){
//separa o conteÃºdo da linha por ' : ' IP : Usuario : Porta
$conteudo = explode(" : ", $linha[$i]);
//monta linha da tabela
echo
"<tr><td>".$conteudo[0]."</td><td>".$conteudo[1]."</td><td>".$conteudo[2]."</td></tr>";
}
echo "</table>";
echo "</div>";
echo "</div>";
echo "</body>"."\n";
echo "</html>"."\n";
```

```
?>
```

Arquivos Exemplos:

/var/www/arquivos/firewall

```
#Define politica padrÃ£para INPUT e FORWARD como negado
#Limpra Todas as Regas
iptables -F
iptables -t nat -F
#Permite saida do firewall
iptables -P INPUT DROP
iptables -P FORWARD DROP
iptables -P OUTPUT ACCEPT


#Libera Acesso SSH
iptables -A INPUT -p TCP --dport 22 -j ACCEPT
#Libera resposta de PING
iptables -A INPUT -p ICMP -j ACCEPT
#Libera Acesso na Porta 80
iptables -A INPUT -p TCP --dport 80 -j ACCEPT
#Libera Acesso na Porta 443
iptables -A INPUT -p TCP --dport 443 -j ACCEPT
#Libera FWD da Porta 3389
iptables -A FORWARD -i eth1 -o eth0 -p tcp --sport 3389 -j ACCEPT
iptables -t nat -A PREROUTING -p tcp --dport 3389 -i eth0 -j DNAT --to 192.168.0.2
```

/var/www/arquivos/firewallscript

```
#Eliminar entradas com mais de 6 horas.

#Adiciona os IPS nas regas
while IFS=" : " read ip user porta data
do
#echo iptables -A FORWARD -i eth0 -o eth1 -p tcp -s $ip/255.255.255.255 --dport $porta -j
ACCEPT
iptables -A FORWARD -i eth0 -o eth1 -p tcp -s $ip/255.255.255.255 --dport $porta -j
ACCEPT
echo "OK"
done < /var/www/arquivos/users_ip
```

/var/www/arquivos/users

```
admin : 21232f297a57a5a743894a0e4a801fc3 : 9901
jm : 3da770cc56ed4407b6aaf10ad4e72b4d : 3389
```

/var/www/arquivos/user_ip

```
10.0.0.53 : admin : 9901 : 26/10/2012 - 18:05
```