

**Carlos Humberto Ribeiro**

**Implantação de um sistema integrado para autenticação segura de usuários  
e permissão de acesso em um proxy transparente**

Monografia de Pós-Graduação “*Lato Sensu*”  
apresentada ao Departamento de Ciência da  
Computação para obtenção do título de Especialista  
em “Administração em Redes Linux”

Orientador  
Prof. Msc. Denilson Vedoveto Martins

Lavras  
Minas Gerais - Brasil  
2007



**Carlos Humberto Ribeiro**

**Implantação de um sistema integrado para autenticação segura de usuários  
e permissão de acesso em um proxy transparente**

Monografia de Pós-Graduação “*Lato Sensu*”  
apresentada ao Departamento de Ciência da  
Computação para obtenção do título de Especialista  
em “Administração em Redes Linux”

*Aprovada em 22 de Setembro de 2007*

---

Professor Msc. Sandro Melo

---

Professor Esp. Samuel Pereira Dias

---

Prof. Msc. Denilson Vedoveto Martins  
(Orientador)

Lavras  
Minas Gerais - Brasil  
2007



*Dedico este trabalho aos meus Pais, Adercides Ribeiro e Marlúcia Magnabosco,  
à minha esposa, Eliane . Espero que um dia eu consiga lhes dar uma justa  
compensação por minhas ausências , para conseguir terminar este Curso.*



## **Agradecimentos**

Agradeço especialmente aos meus Pais, Adercides e Marlúcia, à minha esposa, Eliane, que é essencial na minha vida profissional, desde plantar a semente até a estimular a colher os frutos, ao meu Orientador, Denilson, e aos demais professores do curso.





# Sumário

<b>1</b>	<b>Introdução</b>	<b>1</b>
1.1	Objetivo Geral . . . . .	1
1.2	Objetivos Específicos . . . . .	2
1.3	Estrutura do Trabalho . . . . .	3
<b>2</b>	<b>Revisão da Literatura</b>	<b>5</b>
2.1	Visão geral de controle de acesso à <i>Internet</i> , usando um <i>firewall/gateway</i> autenticado . . . . .	5
2.2	Filtro de pacotes <i>Iptables</i> . . . . .	5
2.3	Sistema <i>cache</i> e <i>proxy</i> transparente . . . . .	8
2.4	Linguagem <i>Perl</i> . . . . .	10
2.5	Os módulos <i>CPAN</i> da Linguagem <i>Perl</i> . . . . .	10
2.6	Banco de Dados <i>MySQL</i> . . . . .	11
2.7	Modo de Administração <i>MySQL</i> através do <i>PhpMyAdmin</i> . . . . .	12
2.8	Utilização do <i>GnuPg</i> como alternativa de criptografia <i>PGP</i> . . . . .	12
2.9	Servidor <i>web Apache</i> . . . . .	13
2.10	Servidor <i>web Apache</i> com conexão segura com chaves ( <i>SSL</i> ) . . . . .	15
2.11	Considerações Finais . . . . .	16
<b>3</b>	<b>Materiais e Métodos</b>	<b>17</b>
3.1	Materiais . . . . .	17
3.2	Metodologia . . . . .	18
3.3	Instalações e Configurações . . . . .	19
3.3.1	O sistema <i>NoCatAuth</i> . . . . .	19
3.3.2	Pré-requisitos para instalação do sistema <i>NoCatAuth</i> . . . . .	22
3.3.3	Instalação dos Módulos Adicionais <i>Perl</i> . . . . .	22
3.3.4	Instalação do Sistema . . . . .	24
3.3.5	Criação de chaves criptografadas em <i>GnuPg</i> . . . . .	26

3.3.6	Criação Certificados Digitais usados pelo servidor <i>web Apache</i> . . . . .	26
3.3.7	Configuração do servidor <i>web Apache</i> e do certificado digital	28
3.3.8	Configuração dos parâmetros do sistema <i>NoCatAuth</i> . . . . .	29
3.3.9	Importação da Tabela de nome <i>nocat</i> do Banco de dados <i>MySQL</i> . . . . .	34
3.3.10	Ajustes Finais de atributos relacionados aos arquivos . . . . .	34
3.3.11	Configuração do sistema de filtro de pacotes <i>Iptables</i> . . . . .	35
3.3.12	Arquivo de inicialização do sistema <i>NoCatAuth</i> . . . . .	36
3.3.13	Considerações Finais . . . . .	37
<b>4</b>	<b>Funcionamento do sistema de autenticação <i>NoCatAuth</i></b>	<b>39</b>
4.1	Telas padrões do sistema <i>NoCatAuth</i> . . . . .	40
4.2	Alterações e implementações nas telas do sistema <i>NoCatAuth</i> . . . . .	43
4.3	Considerações Finais . . . . .	45
<b>5</b>	<b>Conclusões e trabalhos futuros</b>	<b>47</b>
	<b>Referências Bibliográficas</b>	<b>50</b>
<b>A</b>	<b>Instalação do servidor <i>web Apache</i></b>	<b>51</b>
A.1	Verificando a existência do servidor <i>web Apache</i> . . . . .	51
A.2	Instalando o servidor <i>web Apache</i> . . . . .	51
A.3	Iniciando o servidor <i>web Apache</i> . . . . .	52
<b>B</b>	<b>Configuração do Servidor de <i>Dhcp</i></b>	<b>53</b>
<b>C</b>	<b>Instalação do módulo de acesso a banco de dados <i>MySQL</i> em modo gráfico <i>PhpMyAdmin</i></b>	<b>55</b>
<b>D</b>	<b>Instalação e configuração do servidor de banco de dados <i>MySQL</i></b>	<b>57</b>
D.1	Verificando a existência do servidor de banco de dados <i>MySQL</i> . . . . .	58
D.2	Instalando o servidor de Banco de dados <i>MySQL</i> . . . . .	58
D.3	Iniciando o servidor de banco de dados <i>MySQL</i> e configurações Adicionais . . . . .	58

# Lista de Figuras

2.1	Tráfego dos pacotes e tomada de decisão através de regras <i>Iptables</i> (FILHO, 2004) . . . . .	6
2.2	Rota de pacotes através dos serviços de NAT <i>Network Address Translation</i> (FILHO, 2004) . . . . .	7
2.3	Rota do pacote através das tabela <i>Filter</i> do <i>Iptables</i> (FILHO, 2004)	8
2.4	Gráfico de utilização do servidor <i>web Apache</i> (NETCRAFT, 2007) .	14
3.1	Situação atual do sistema de autenticação da instituição . . . . .	19
3.2	Captura de tráfego pelo sistema <i>NoCatAuth gateway</i> (BRANDÃO, 2004) . . . . .	19
3.3	Estrutura física de rede para o funcionamento correto do sistema <i>NoCatAuth gateway</i> (BRANDÃO, 2004) . . . . .	20
3.4	Características do sistema <i>NoCatAuth gateway</i> (BRANDÃO, 2004)	21
3.5	Instalação dos módulos adicionais e da biblioteca CPAN (BRANDÃO, 2004) . . . . .	23
3.6	Verificação dos módulos instalados (BRANDÃO, 2004) . . . . .	24
3.7	Obtenção via <i>wget</i> e extração dos arquivos de configuração (BRANDÃO, 2004) . . . . .	24
3.8	Obtenção do arquivo de instalação via <i>download</i> . . . . .	25
3.9	Alterações arquivo <i>detect-fw.sh</i> para o funcionamento <i>Firewall 2.6</i> - sistema <i>Fedora Core 6.0</i> . . . . .	25
3.10	Instalação do sistema <i>NoCatAuth</i> no modo <i>gateway</i> e autenticação (BRANDÃO, 2004) . . . . .	26
3.11	Criação de chaves criptografadas em <i>GnuPg</i> (BRANDÃO, 2004) . .	26
3.12	Processo de criação de chaves criptografadas com <i>GnuPg</i> com recebimento de dados . . . . .	27
3.13	Criação do diretório para armazenamento de certificados digitais (BRANDÃO, 2004) . . . . .	27
3.14	Geração dos certificados digitais (BRANDÃO, 2004) . . . . .	28

3.15	Configuração dos parâmetros para <i>CGI-bin</i> no diretório de instalação <i>NoCatAuth</i> (BRANDAO, 2004) . . . . .	28
3.16	Configuração do diretório de execução do servidor <i>web Apache</i> com redirecionamento para a porta padrão do sistema <i>NoCatAuth</i> .	29
3.17	Modificação no arquivo de configuração <i>ssl.conf</i> para a leitura da chave de certificação digital do servidor <i>web Apache</i> . . . . .	30
3.18	Configuração dos parâmetros usados pelo servidor de autenticação internos ao arquivo <i>nocat.conf</i> . . . . .	31
3.19	Configuração dos parâmetros usados pelo servidor de <i>gateway</i> internos ao arquivo <i>nocat.conf</i> . . . . .	32
3.20	Criação do banco de dados <i>MySQL</i> denominada <i>nocat</i> responsável pelo armazenamento dos usuários do sistema . . . . .	34
3.21	Importação da tabela de dados <i>MySQL</i> denominada <i>nocat</i> responsável pelo armazenamento dos usuários do sistema (BRANDAO, 2004)	34
3.22	Redirecionamento das requisições realizadas no servidor de <i>proxy</i> , <i>Squid</i> . . . . .	36
3.23	Redirecionamento padrão para requisições realizadas no sistema de autenticação <i>NoCatAuth</i> . . . . .	36
3.24	Arquivo de Inicialização do sistema <i>NoCatAuth</i> . . . . .	37
4.1	Funcionamento do Sistema de Autenticação <i>NoCatAuth</i> . . . . .	39
4.2	Tela de autenticação de usuários no sistema <i>NoCatAuth</i> . . . . .	40
4.3	Conexão Segura, com confirmação do certificado digital . . . . .	41
4.4	Tela de <i>Splash</i> contendo tempo de expiração e tela de boas-vindas	41
4.5	Registro de usuários . . . . .	42
4.6	Tela inicial do sistema <i>NoCatAuth</i> , ressaltando botão de <i>Logout</i> , informações ao usuário e padronização. . . . .	43
4.7	Tela de boas-vindas ao sistema de autenticação contendo a esquerda tela de <i>Splah</i> , com botão de <i>Logout</i> . . . . .	44
4.8	Tela de segurança do sistema de autenticação <i>NoCatAuth</i> . . . . .	45
B.1	Estrutura de uma rede com servidor <i>Dhcp</i> (SUP, 2004) . . . . .	53
B.2	Configuração de um servidor <i>Dhcp</i> através do arquivo <i>/etc/dhcpd.conf</i>	54
C.1	Instalação do módulo de administração de banco de dados <i>PhpMyAdmin</i> . . . . .	55
C.2	Tela com funcionalidades do sistema <i>PhpMyAdmin</i> . . . . .	56
D.1	Instalação do servidor de banco de dados <i>MySQL</i> através do utilitário <i>yum</i> . . . . .	58

# Lista de Tabelas

2.1	Resultados da pesquisa da <i>NetCraft</i> Agosto de 2007 da Utilização de servidores <i>web</i> . . . . .	13
3.1	Pré-requisitos do <i>Sistema</i> . . . . .	22



## **Resumo**

Este trabalho apresenta uma proposta para implantação e melhoria na solução para controle de acesso autenticado à *Internet* usando a solução denominada *NoCatAuth*. As alterações propostas envolvem melhorias no processo de controle do acesso à *Internet* dos usuários registrados no servidor *MySQL*, através de mudanças realizadas nos formulários escritos em *HTML* com *Scripts* em linguagem *Perl* e que permite o uso do *Squid* em modo transparente, sem os inconvenientes da autenticação a cada nova instância aberta do navegador da *Internet*.

**Palavras-Chave:**

# Capítulo 1

## Introdução

Atualmente, os sistemas computacionais estão cada vez mais integrados à *Internet*, tornando os usuários destes sistemas cada vez mais dependentes deste recurso, tanto no trabalho como em casa. Por isso, muitas pessoas, em alguns casos, abusam e fazem mal uso deste recurso no ambiente de trabalho.

Quando se fala em conceder acesso à *Internet* para os funcionários de uma empresa, a primeira preocupação é com relação aos abusos cometidos, obrigando a empresa a tomar algumas medidas para o controle de acesso. Ao invés da utilização da *Internet* como instrumento de trabalho, ou para obtenção de informações necessárias à execução de tarefas, pode acontecer que a mesma seja usada para navegar por *sites* com conteúdos impróprios, obtenção e instalação de programas piratas, além de músicas, fotos e outros tipos de entretenimento, que podem, ou não serem utilizados no ambiente de trabalho. Esta situação coloca em risco os sistemas e as informações da empresa, pois aumenta a possibilidade de que, por exemplo, os computadores sejam infectados por um vírus, gerando assim, além do consumo de banda pela navegação, perdas financeiras e desperdício de tempo de trabalho do funcionário.

Torna-se necessário o controle de alguma forma do acesso concedido aos usuários à *Internet*, e este trabalho busca alternativas para implementação deste controle de acesso, via autenticação de usuários, concedendo somente a usuários autenticados o acesso à *Internet*.

### 1.1 Objetivo Geral

Este trabalho tem como objetivo geral propor um implementação de um sistema de autenticação segura de usuários, utilizando de *proxy* transparente para acesso à *Internet*, com utilização criptografia na comunicação entre o servidor de auten-



ticação e o cliente. O armazenamento dos dados pertinentes a cada usuário serão mantidos em banco de dados *MySQL*, com senhas criptografadas. Desta forma, serão utilizadas somente aplicações livres, sem nenhum custo.

A motivação para implantação de um sistema de autenticação, veio da necessidade de controle dos usuários em uma rede sem fio, permitindo assim a conexão apenas a usuários previamente cadastrados. Grande parte do tempo do projeto caracterizou-se pela busca de soluções adequadas para implantação de um sistema de autenticação, que atendesse e implementassem com um nível de segurança uma rede sem fio, na instituição da Prefeitura Municipal de Sacramento. Após várias pesquisas e levantamento dos preços para a implantação da solução, o sistema de autenticação *NoCatAuth*, surgiu como uma das possíveis soluções, tornando necessárias algumas modificações, para que a solução se mostrasse viável.

## 1.2 Objetivos Específicos

- Configuração e instalação dos módulos adicionais da linguagem *Perl*,<sup>1</sup> *DBI*, *DBD::Mysql*, *Digest::MD5* e *Net::Netmask*.
- Instalação do sistema de autenticação de usuários utilizando a solução *NoCatAuth*<sup>2</sup>.
- Criação de chaves criptografadas utilizando *GnuPG*<sup>3</sup> para comunicação com o servidor de autenticação.
- Criação de certificados digitais para acesso seguro ao servidor *web Apache*, garantindo uma comunicação segura entre o servidor de autenticação e o cliente.
- Configuração do servidor *web Apache*<sup>4</sup> para utilização de certificado digital.
- Configuração do sistema de autenticação de usuários *NoCatAuth*.
- Importação do banco de dados *MySQL*<sup>5</sup> responsável em armazenar dados dos usuários do sistema.
- Criação e configuração do arquivo de inicialização do sistema de autenticação de usuários.

---

<sup>1</sup><http://www.perl.org/>

<sup>2</sup><http://www.nocat.net/>

<sup>3</sup><http://www.gnupg.org/>

<sup>4</sup><http://www.apache.org/>

<sup>5</sup><http://www.mysql.com/>

- Ajustes no sistema de filtro de pacotes *Iptables*<sup>6</sup> para o funcionamento do sistema de autenticação.
- Adequação das telas de recebimento de informações com padrões específicos da empresa de implantação.
- Criação do botão de *Logout* na tela principal para o desligamento do usuário do sistema de autenticação.
- Criação de uma tela de segurança no acesso para registro de novos usuários, bloqueando ao usuário comum este recurso.

### 1.3 Estrutura do Trabalho

O presente trabalho está dividido em cinco capítulos, incluindo esta introdução. O restante está organizado como segue.

No Capítulo 2, são apresentadas definições sobre os sistemas *proxy* e como são utilizados para controlar o acesso à *Internet*. Também são apresentados os tipos de sistema *proxy* que podem ser implementados como o popular *software Squid*<sup>7</sup> e as vantagens e limitações deste sistema como um sistema *proxy* transparente. Também são apresentadas algumas características de recursos existentes no *kernel* do *Linux* e oferecidos pelo filtro de pacotes *Iptables*, os quais também podem ser usados para implementar o conceito de *proxy* transparente. Este capítulo também apresenta os pré-requisitos para instalação do sistema de autenticação de usuários *NoCatAuth*, que aliado ao *proxy* transparente libera o acesso aos recursos da *Internet*.

No Capítulo 3, é tratada a implantação e características do software escrito em linguagem *Perl*, denominado *NoCatAuth* para o controle do acesso a *Internet* com *proxy* transparente através de uma autenticação de usuários. São abordados os pontos necessários para configuração do sistema tornando o mesmo operacional e pronto para o uso. Este capítulo também trata de geração de chaves seguras com *Openssl*<sup>8</sup> para certificação digital através do uso do módulo *Apache* garantindo assim uma segurança adicional na autenticação do usuário, além da geração de chave através de *GnuPg* como módulo de segurança adicional ao sistema de autenticação.

As alterações no sistema *NoCatAuth* são propostas no Capítulo 4. O capítulo mostra de uma maneira clara, as modificações e adequações realizadas no sistema

---

<sup>6</sup><http://www.netfilter.org/projects/Iptables/index.html>

<sup>7</sup><http://www.squid-cache.org/>

<sup>8</sup><http://www.openssl.org/>

de autenticação *NoCatAuth*. As alterações propostas neste módulo partem inicialmente das modificações das telas que o sistema disponibiliza, gerando novas funcionalidades. São acrescentadas funcionalidades como a possibilidade de *logout* do usuário na tela principal, a proteção ao acesso do registro de novos usuários, impossibilitando assim que um usuário consiga se registrar sem o consentimento do administrador do sistema, além de modificações estéticas.

O Capítulo 5 apresenta algumas considerações a respeito do controle de acesso à *Internet* em relação aos sistemas operacionais em geral e em relação ao *Linux*, acentuando as vantagens deste, bem como, sobre cuidados que devem ser tomados para sua implementação. São descritas em detalhes as melhorias que o sistema possibilita dentro de uma rede privada ou rede sem fio para o controle de acesso à *Internet*, usando recursos de usuários autenticados. Este capítulo ainda apresenta os trabalhos futuros, provenientes deste sistema de autenticação de usuários.

## Capítulo 2

# Revisão da Literatura

### 2.1 Visão geral de controle de acesso à *Internet*, usando um *firewall/gateway* autenticado

Através de serviços de *proxy* e filtragem de pacotes torna-se possível o controle de acesso à *Internet*. A utilização de um *gateway* facilita a gerência de acesso à *Internet* e permite maximizar o aproveitamento da conexão disponível, onde o *gateway* torna-se responsável em prover acesso à *Internet* às demais máquinas da rede interna. O *gateway* também contribui para incrementar uma segurança de rede interna contra ataques oriundos da *Internet*, onde todos os esforços de prevenção e defesa em vários níveis são concentrados no mesmo.

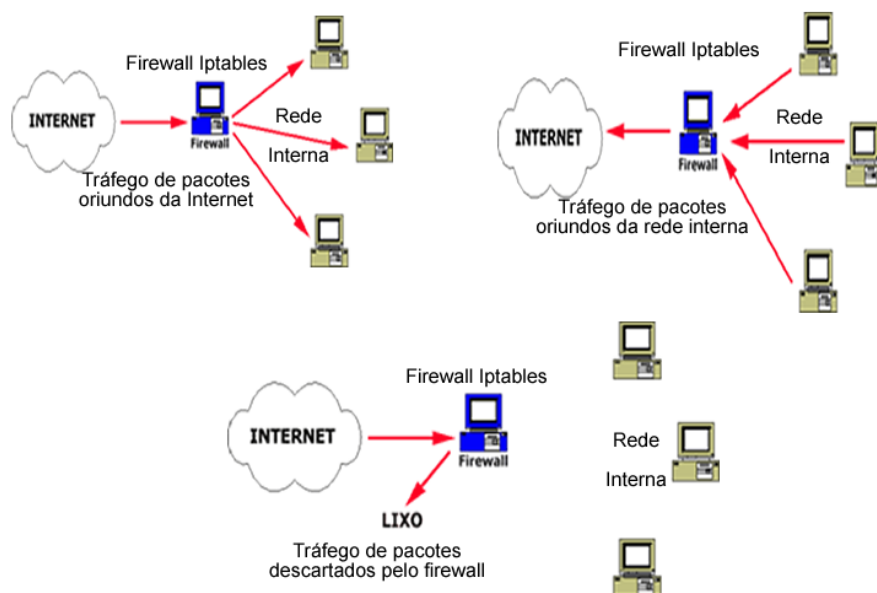
Os aplicativos utilizados para filtragem de pacotes e conteúdos em sistemas *Linux*, é a dupla *Squid* e *Iptables*. No ponto de autenticação de usuários existem várias formas que podem ser usadas, onde a forma usual é através do próprio *Squid*, seguindo os padrões do programa. Este sistema de autenticação obriga o usuário a se identificar a cada vez que o navegador acessar a *Internet* ou uma nova instância for criada. Outra solução mais robusta trata-se do sistema de autenticação de usuários *NoCatAuth*, que trabalha ligado ao *Squid*, liberando apenas as requisições autenticadas, gerando assim um desconforto menor ao usuário que utiliza do sistema.

### 2.2 Filtro de pacotes *Iptables*

Segundo (FILHO, 2004), o *firewall* é um programa que tem como objetivo proteger a rede local, através do bloqueio do tráfego indesejado, evitando assim que dados corporativos de uma instituição ou mesmo dados pessoais possam ser violados.

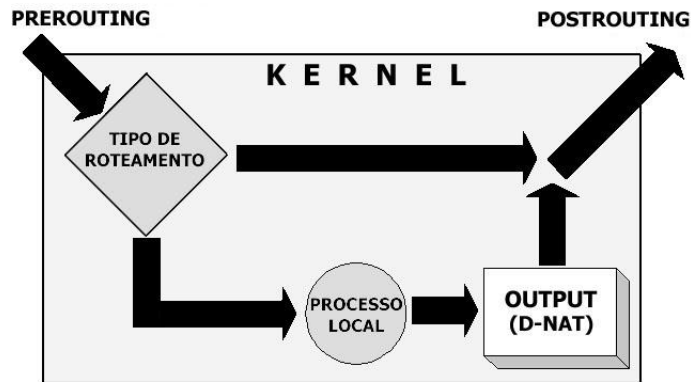
O *iptables* é um *firewall* em nível de pacotes e estados que opera basicamente na camada 2 da pilha de protocolos *TCP/IP*, que por definição somente são capazes de filtrar pacotes com base em informações do cabeçalho *IP*. Conforme (ANDREASSON, 2005), o *iptables* possui recursos que o permitem trabalhar também na camada 3 da pilha de protocolos *TCP/IP*, com informações existentes nos cabeçalhos de protocolos como o *UDP* ou *TCP*, em camada inferior, com endereços *MAC* e também em relação ao estado das conexões.

O *Iptables* possui a capacidade de analisar cabeçalhos (*Headers*) de pacotes através da comparação de regras enquanto os mesmos trafegam, onde através de uma extensa comparação de regras previamente adicionadas, decide se um pacote tem ou não permissão de acesso, conforme pode-se observar na figura 2.1. Em *firewalls* mais restritivos, o pacote é bloqueado e registrado, para que o administrador tenha conhecimento sobre o que está acontecendo em seu sistema.



**Figura 2.1:** Tráfego dos pacotes e tomada de decisão através de regras *Iptables* (FILHO, 2004)

O *Iptables*, além de filtrar pacotes, oferece serviços de *NAT* (*Network Address Translation*), usados para alterar os endereços de origem ou destino dos pacotes enviados (*SNAT* e *DNAT*), conforme pode-se observar na figura 2.2. Esta mudança de endereços de origem ou destino dos pacotes dependem da capacidade que o *Linux* possui para rastreamento das conexões, e da verificação dos pacotes que pertencem a um mesmo fluxo de dados “*connection tracking*”.



**Figura 2.2:** Rota de pacotes através dos serviços de NAT Network Address Translation (FILHO, 2004)

Quando um pacote é recebido por uma máquina, possuindo o *firewall Iptables* em execução, ele percorre vários passos ou regras, até chegar à aplicação ou ser reencaminhado para outra máquina. Em cada etapa percorrida pelo pacote, poderão ser definidas regras com ações a serem tomadas, com base em suas características, em comparação à valores pré-definidos em cada regra. Estas ações são classificadas em tabelas, de acordo com a finalidade da ação: *mangle*, *nat*, *filter* (a tabela padrão), sendo a classe mais utilizada de *firewall* (NETO, 2004). As principais ações são: *DROP*, *ACCEPT*, *REDIRECT*, *REJECT*, *LOG*, *DNAT*, *SNAT*, *MASQUERADE*, destacando-se neste trabalho, com base em seus objetivos, a *chain PREROUTING*, na tabela *nat*, com as ações *DNAT*, *ACCEPT*, *DROP*.

Pela *chain FORWARD* passam os pacotes que serão reencaminhados pelo *kernel* do *Linux* para outra rede, no caso em questão neste trabalho, a *Internet*. As regras desta *chain* na tabela *filter* determinam a quais pacotes será permitido o roteamento (*ACCEPT*) e quais serão eliminados (*DROP*), conforme pode-se observar na figura 2.3. Esta escolha pode ser feita com base no protocolo e porta usados, endereço *IP* de origem ou destino do pacote, dentre várias outras opções.

A regra *POSTROUTING* é a última *chain* que um pacote passa ao sair da máquina rumo à *Internet*. A tabela *NAT*, através da ação *SNAT*, contém as regras responsáveis em definir como será alterado o endereçamento de *IP* da rede interna para um endereço *IP* válido. A ação *MASQUERADE* é utilizada para alterar o endereço de origem do pacote para um endereço dinâmico, sendo configurado na *interface* de rede da *Internet*, que é consultada a cada início de conexão. Já o *SNAT* é usado para definir um *IP* fixo para alteração do endereço de origem do pacote. Um registro de log pode ser feito, usando as ações *LOG* ou *ULOG*, antes das alterações do endereço de origem.

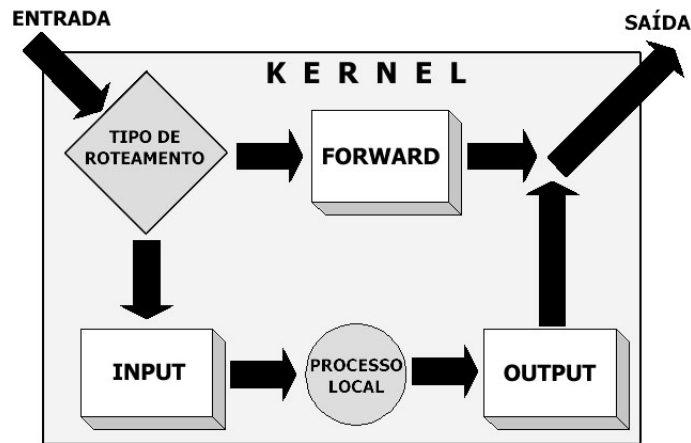


Figura 2.3: Rota do pacote através das tabela *Filter* do *Iptables* (FILHO, 2004)

## 2.3 Sistema *cache* e *proxy* transparente

Segundo (MARCELO, 2005), um *proxy cache* é um servidor que tem o objetivo de acelerar o acesso a um determinado conteúdo oriundo da *Internet*. Esta aceleração é feita com a criação de um *cache* local para onde são enviadas cópias do conteúdo requisitado. O *Squid* oferece o serviço de *cache*. No caso do serviço de *HTTP*, quando um cliente em uma estação específica solicitar ao servidor *proxy* um pedido para baixar uma página, o navegador realiza o pedido ao servidor *proxy*. Este servidor tem armazenadas as páginas visitadas recentemente e consulta este *cache* de páginas averiguando se a página web solicitada está disponível. Caso a página esteja disponível, esta página armazenada no *cache*, é enviada ao cliente. caso não esteja no *cache*, ou se a versão existente for muito antiga, o servidor *proxy web* busca a mesma do *site* em questão, armazenando a página no seu *cache* e posteriormente enviando à estação de trabalho.

“O *Squid* é, ao mesmo tempo, um servidor *proxy* e de *caching* que roda sob o *Linux* e suporta diversos protocolos, entre os quais *HTTP*, *FTP* e *SSL*” (NEMETH; SNYDER; HEIN, 2004). Embora o nome *Squid* apresenta-se frequentemente associado ao termo *proxy*, isto não é absolutamente correto. Conforme pode-se observar em (UCHÔA, 2005), um *proxy* é um software que atua como ponto entre duas redes, controlando o tráfego de acordo com seu conteúdo, ou ainda, um serviço intermediário entre um cliente e o servidor real dos serviços.

O servidor de *proxy* utilizando *cache* aumenta a eficiência na utilização da rede, porque a página da *web* é carregada imediatamente para o cliente a partir do servidor *proxy*. O acesso à *Internet* pelo *proxy* é realizado apenas quando a página

solicitada pelo usuário, encontra-se desatualizada no servidor de *cache*. (STANGER; LANE, 2002).

Quando o *Iptables* faz SNAT, efetivamente ocorre um serviço de *proxy* transparente, trabalhando na camada 2 da pilha de protocolos *TCP/IP*, pois endereços *IP* da rede interna estão sendo alterados para alcançar a rede externa. De forma inversa, uma vez estabelecida a conexão com a *Internet*, o mecanismo de (*connection tracking*) tornará possível que os pacotes que retornarem da rede externa recebam os endereços das respectivas máquinas às quais se destinam. “*O NAT é, na verdade, um proxy fundamental: um único host faz as solicitações em nome de todos os hosts internos, ocultando assim suas identidades da rede pública.*” (STREBE; PERKINS, 2002)

A utilização do *Squid* como servidor de *proxy* baseia-se em duas formas de configuração: como *proxy* “convencional” ou como *proxy* “transparente”. Por padrão, o serviço responde em uma porta específica no servidor (porta TCP 3128), mas que pode ser modificado de acordo com as necessidades do administrador do sistema. Os aplicativos nas estações de trabalho devem ser configurados para explicitamente acessar os serviços HTTP e FTP através da porta definida dentro das configurações do servidor *proxy*. A principal desvantagem é que cada aplicativo que deseje utilizar o *proxy* tem de ser configurado desta forma para utilizá-lo. Quando usado desta maneira, o *Squid* pode ser configurado para exigir autenticação do usuário, antes de atender suas requisições, sendo um meio efetivo de controlar quem pode ter acesso ao recurso da *Internet*.

O acesso à serviços específicos na *Internet* passando através de um *proxy*, sem necessidade de efetuar qualquer alteração nos aplicativos do usuário é denominado *proxy* transparente, sendo considerado como a forma mais prática de utilizar o *Squid*. Através de uma regra no *Iptables* as requisições realizadas pelos clientes são redirecionadas para as portas do serviço que o *proxy* responde, sendo atendidas de modo invisível ao usuário. Assim, as requisições dos clientes sempre serão direcionadas ao servidor *proxy*, resolvendo o problema anteriormente citado, quando os clientes conseguem contornar o sistema *proxy* e acessar diretamente o serviço na *Internet*.

Segundo (VISOLVE, 2006), a grande desvantagem do uso de autenticação interna ao *Squid* é que não é possível ativar o recurso de autenticação de usuários aliado ao uso de *Proxy* transparente. Esta desvantagem é a maior motivadora da realização deste trabalho, e de outros já existentes, como por exemplo o *NoCatAuth*<sup>1</sup> e *natacl*.<sup>2</sup>

---

<sup>1</sup><http://www.nocat.net>

<sup>2</sup><http://natacl.sourceforge.net/>



A utilização de um sistema de *proxy* através do *Squid* permite ao administrador regras de bloqueio de palavras e filtro de conteúdo inadequado, além do bloqueio de *sites* específicos, de acordo com seu endereço ou domínio. Todas as requisições são registradas em *logs* e ferramentas para analisar estes arquivos de *log* podem ser utilizadas. Através da análise do arquivo de *log* `access.log` do *Squid*, quando configurado para exigir autenticação, pode-se obter diversas informações, tais como: hora de acesso, *sites* acessados, quantidade de conexões, falhas de conexão entre outras informações.

O *SARG* (*Squid Analysis Report Generator*)<sup>3</sup> é uma destas ferramentas que, conforme definição existente em seu *site*, “*permite ver onde seus usuários estão indo na Internet*”.

## 2.4 Linguagem *Perl*

“A linguagem *Perl*<sup>4</sup> é gratuita e foi criada por Larry Wall, em um esforço para produzir relatórios para um sistema de informações de erros. Larry construiu uma nova linguagem de *script* para esta finalidade e então lançou-a na Internet, pensando que alguém mais poderia achá-la útil. No espírito do *software livre*, outras pessoas sugeriram melhorias e ainda maneiras de implementá-las e o *Perl* transformou-se de uma linguagem de *scripts* atraente para uma linguagem de programação completa”. (SIELVER, 1999)

Atualmente, *Larry* faz pouco desenvolvimento por si só, mas ele é o cabeça de uma equipe de desenvolvimento básico conhecida como *Perl Posters*. Os *Posters* determinam quais novos recursos deverão ser adicionados e quais erros indesejáveis deverão ser corrigidos.

## 2.5 Os módulos *CPAN* da Linguagem *Perl*

“A *CPAN*<sup>5</sup> representa os interesses de desenvolvimento de uma seção cruzada da comunidade *Perl*. Ela contém os utilitários *Perl*, módulos, documentação e claro, a própria distribuição *Perl*. A *CPAN* foi criada por Jarkko Hietaniemi e Andreas König.” (SIELVER, 1999)

O sistema pessoal da *CPAN* é o `funet.fi`, mas esta é também espelhada em muitos outros sistemas em todo o globo. Isto assegura que qualquer pessoa com uma conexão da *Internet* a qualquer momento poderá ter um acesso confiável ao

---

<sup>3</sup><http://sarg.sourceforge.net/>

<sup>4</sup><http://www.perl.org/>

<sup>5</sup><http://www.perl.com/CPAN>

conteúdo da *CPAN*. Como a estrutura de todos os sites *CPAN* é a mesma, um usuário que pesquisa a versão atual da linguagem *Perl* poderá estar certo de que o arquivo `latest.tar.gz` é o mesmo em todo o site.

## 2.6 Banco de Dados *MySQL*

*“O MySQL é um servidor e gerenciador de banco de dados SGDB relacional, de licença dupla (sendo uma delas software livre), projetado inicialmente para trabalhar com aplicações de pequeno e médio portes, mas hoje atendendo a aplicações de grande porte e com mais vantagens do que seus concorrentes. Possui todas as características que um banco de dados de grande porte precisa, sendo reconhecido por algumas entidades como banco de dados open source com maior capacidade para concorrer com programas similares de código fechado, tais como SQL Server da instituição Microsoft <sup>6</sup> e Oracle <sup>7</sup>”.* (MILANI, 2006)

O *MySQL* teve origem quando os desenvolvedores *David Axmark*, *Allan Larson* e *Michael Monty Widenius* na década de 90, precisaram de uma interface *SQL* compatível com as rotinas *ISAM* que utilizavam em suas aplicações e tabelas. Em um primeiro momento, tentaram utilizar a *API mSQL*. Contudo a *API* não era tão rápida quanto eles precisavam, pois as rotinas utilizadas eram de baixo nível (mais rápidas que rotinas normais). Utilizando a *API* do *mSQL*, escreveram em linguagem *C* e *C++* uma nova *API* que deu origem ao *MySQL*.

Com o ótimo resultado gerado por essa nova *API*, o *MySQL* começou a ser difundido e seus criadores fundaram a instituição responsável por sua manutenção que é a *MySQL AB* <sup>8</sup>.

A partir dessa fase, o *MySQL* tornou-se mais conhecido por sua característica acesso rápido, sendo cada vez mais utilizado. Novas versões foram lançadas, contemplando novas necessidades e firmando, assim, sua posição no mercado. Sua mais recente versão é a 5.1, a qual conta com novos recursos, estabelecendo sua capacidade para competir com sistemas gerenciadores de bancos de dados proprietários de maior popularidade.

---

<sup>6</sup><http://www.microsoft.com>

<sup>7</sup><http://www.oracle.com>

<sup>8</sup><http://www.mysql.com/>

## 2.7 Modo de Administração *MySQL* através do *PhpMyAdmin*

Segundo (MILANI, 2006), o *PhpMyAdmin*<sup>9</sup> é uma ferramenta de uso gráfico simples e prática escrita em linguagem *PHP* para auxiliar no gerenciamento e administração do servidor contendo bases com dados *MySQL*.

Por meio de uma interface gráfica *web*, disponibiliza alguns dos recursos mais utilizados no *MySQL*, como a criação de bases de dados, criação e manutenção de tabelas, gerenciamento de processos, de forma rápida e intuitiva para o usuário.

O *PhpMyAdmin* apenas automatiza algumas das linhas de comandos mais frequentes do *MySQL*. Não existe nenhum recurso exclusivo do *PhpMyAdmin* que não possa ser feito a partir do terminal cliente do *MySQL*, sendo que algumas das funcionalidades obtidas pelo terminal não poderão ser acessadas pelo *PhpMyAdmin*.

## 2.8 Utilização do *GnuPg* como alternativa de criptografia *PGP*

Segundo (SIELVER, 1999) o *GNU Privacy Guard (GnuPg)* é uma alternativa de software livre para a suíte de criptografia *PGP*, liberado sob a licença *GNU General Public License*<sup>10</sup>. Ele é parte do projeto *GNU* da *Free Software Foundation*<sup>11</sup> e recebe a maior parte do seu financiamento do governo alemão. O *GnuPg* é completamente compatível com o padrão *IETF*<sup>12</sup> para o *OpenPGP*. As versões atuais do *PGP* (e *Filecrypt* da *Veridis*) são inter-operáveis com o *GnuPg* e outros sistemas compatíveis com o *OpenPGP*. Apesar de algumas versões antigas do *PGP* serem inter-operáveis, nem todas as funcionalidades do novo software são suportadas pelo antigo.

O *GnuPg* é um programa de linha de comando, mas existem vários front-ends que atuam como interfaces gráficas para o *GPG*. Alguns exemplos são:

- *Enigmail*, para o *Mozilla Thunderbird*
- *Seahorse*, baseado no *GNOME*
- *Kgpg*, baseado no *KDE*

---

<sup>9</sup><http://sourceforge.net/projects/phpmyadmin/>

<sup>10</sup><http://www.gnu.org/copyleft/gpl.html>

<sup>11</sup><http://www.fsf.org/>

<sup>12</sup><http://www.ietf.org/>

- *WinPT*, utilitário que funciona na barra de tarefas do sistema do Windows

Com a necessidade da transferência segura de dados considerados confidenciais através das redes públicas, como a *Internet*, o método utilizado, caracteriza-se por uma criação de chaves seguras utilizando o *PGP*<sup>13</sup> que possui licenciamento “pago”. O utilitário *GnuPg* é a substituição gratuita e completa e sem problemas com patentes ou licenciamentos do *PGP*. O *GnuPg* é totalmente compatível com padrão *OpenPGP* que pode ser encontrado na *RFC 2440*<sup>14</sup>. Este utilitário é muito útil na comunicação com outras pessoas que usam o *gpg* ou algum outro aplicativo orientado para *PGP*.

## 2.9 Servidor *web Apache*

O servidor *web Apache* é o servidor com um maior número de utilização em relação aos demais servidores de *Internet* nos dias atuais, de acordo com levantamento da *NetCraft* dos sites *web* ativos em Abril de 2007, conforme pode-se observar na tabela 2.1.

**Tabela 2.1:** Resultados da pesquisa da *NetCraft* Agosto de 2007 da Utilização de servidores *web*.

\* De 127.961.479 sites pesquisados

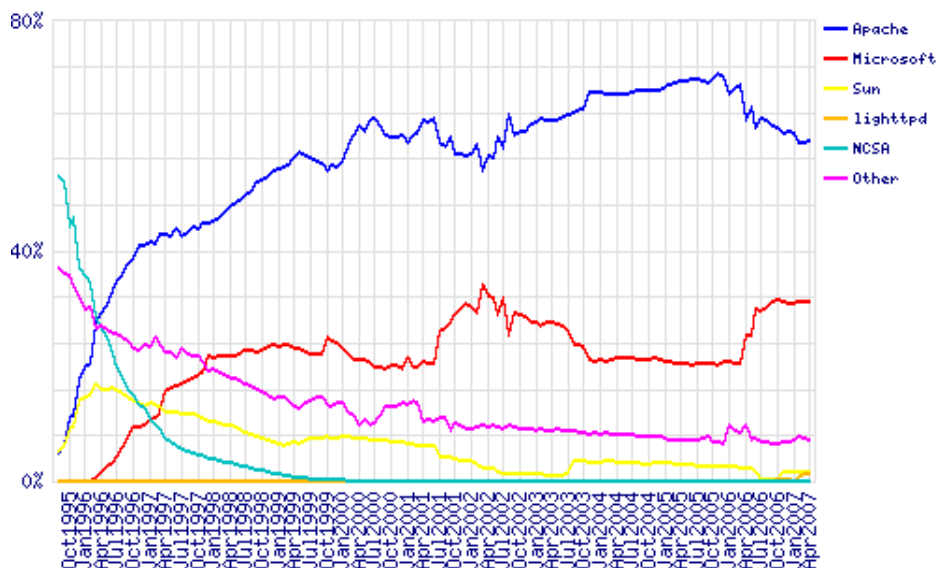
Servidor web	Número	Porcentagem
<i>Apache</i>	65.153.417	50,92
<i>Microsoft ISS</i>	43.861.854	34,28
<i>Google</i>	5.697.504	4,45
<i>Sun</i>	2.195.495	1,72
<i>lighttpd</i>	1.500.126	1,17
<i>Zeus</i>	562.438	0,44

O aumento de interesse no projeto *Apache* nos últimos anos, foi estimulado pelo crescimento do uso de sistemas operacionais livres baseados em *Linux*, que possuem este servidor de *web* como parte integrante de suas instalações. Outro ponto importante para o crescimento da utilização em servidores do servidor *web Apache* é devido, em parte, às falhas de segurança encontradas no *ISS (Internet Information Server)* da *Microsoft* e às vulnerabilidades do *software*. A *IBM* se comprometeu a suportar e usar o servidor *web Apache* como base para suas ofertas relacionadas a *web* e tem dedicado recursos substanciais ao projeto.

O mais revelador é a taxa de uso do servidor *web Apache*, em relação às outras plataformas de servidor em uso pelos desenvolvedores, de acordo com a *NetCraft*, conforme pode-se observar na figura 2.4.

<sup>13</sup><http://www.dca.fee.unicamp.br/pgp/>

<sup>14</sup><http://www.apps.ietf.org/rfc/rfc2440.html>



**Figura 2.4:** Gráfico de utilização do servidor web Apache (NETCRAFT, 2007)

O nome *Apache* apareceu durante o início do desenvolvimento do *software*, porque ele era a “a patchy server” (‘um servidor remendado’), feito de emendas (*patches*) para o código-fonte disponível gratuitamente do servidor web *HTTPd* da *NCSA* (*National Center for Supercomputing Applications*). Por algum tempo, após o projeto do *HTTPd* da *NCSA* ter sido descontinuado, várias pessoas escreveram uma variedade de *patches* para o código, tanto para corrigir erros como para adicionar novos recursos. Grande parte desse código era encontrado na *Internet*, e todos estavam compartilhando livremente, apesar de não haver nenhum gerenciamento. (DUFF, 2003)

Após algum tempo, *Bob Behlendorf* e *Cliff Skolnick* estabeleceram um repositório centralizado desses *patches*, nascendo assim o popular e conhecido *Apache*. Esse projeto ainda é composto de um pequeno grupo básico de programadores, sendo aberto a todos os interessados a enviar *patches* para o grupo, para uma possível inclusão no código.

Em meados de 1999, a *Apache Software Foundation* foi incorporada como uma instituição sem fins lucrativos. Um conselho diretor, eleito anualmente pelos membros da *ASF*<sup>15</sup>, supervisiona a instituição. Essa instituição fornece a base para vários projetos de desenvolvimento de *software* de código-fonte aberto diferentes, incluindo o projeto *Apache web Server*.

<sup>15</sup><http://www.amscan.org/>

## 2.10 Servidor *web Apache* com conexão segura com chaves (SSL)

Segundo (SIELVER, 1999), o *OpenSSL* <sup>16</sup> é uma implementação de código aberto dos protocolos *SSL* e *TLS*. A biblioteca (escrita na linguagem C) implementa as funções básicas de criptografia e disponibiliza várias funções utilitárias. Também estão disponíveis *wrappers* que permitem o uso desta biblioteca em várias outras linguagens.

O *OpenSSL* está disponível para a maioria dos sistemas do tipo *Unix*, incluindo *Linux*, *Mac OS X* e para as quatro versões do *BSD* de código aberto, e também para o *Microsoft Windows*. O *OpenSSL* é baseado no *SSLey* de *Eric Young* e *Tim Hudson*.

A proposta do protocolo *SSL* é permitir a autenticação de servidores, encriptação de dados, integridade de mensagens e, como opção, a autenticação do cliente, operando nas comunicações entre aplicativos de forma interoperável, visando garantir os seguintes objetivos:

- Segurança criptográfica para o estabelecimento de uma ligação segura assim que a conexão entre o servidor e o cliente é estabelecida, assegurando a privacidade na conexão, com a utilização de algoritmos simétricos (como o *DES* ou *RC4*) que negociam uma chave secreta na primeira fase do *handshaking* (usando chaves públicas e assimétricas).
- Autenticação do servidor (e, opcionalmente do Cliente) por meio de algoritmos assimétricos como o *RSA* ou o *DSS*.
- Confiabilidade na conexão, conseguida com o uso de Códigos de Autenticação de Mensagens (*MAC*).

O *SSL* também permite a montagem de um framework onde outras chaves públicas e métodos de encriptação podem ser utilizados, evitando a necessidade de implementação de toda uma pilha de protocolos (com os riscos da vulnerabilidades).

O protocolo *SSL* na sua última versão 3.0 traz como vantagem adicional, o desempenho, reduzindo o número de conexões e minimizando o tráfego na rede podendo ser usado opcionalmente um esquema de cache em memória durante o estabelecimento da sessão, com a finalidade de reduzir o número de conexões e reduzir a atividade no acesso à rede.

---

<sup>16</sup><http://www.openssl.org>

## 2.11 Considerações Finais

Segundo (MARCELO, 2005) o sistema de controle de acesso de usuários à *Internet*, utilizando o *Squid*, ainda não possui nenhuma implementação utilizando autenticação e *proxy* transparente ao mesmo tempo. Hoje, toda instituição que necessite de tais controles aliados, é obrigada a utilizar uma solução de autenticação diferente daquela que o *Squid* possibilita. O *Squid*, implementa as duas soluções para uso, mas infelizmente, em separado, não permitindo assim a sua utilização simultaneamente.

O próximo capítulo apresenta uma solução de autenticação única, utilizando *proxy transparente*, não gerando o desconforto da autenticação de cada instância aberta dentro de um *browser* como ocorre através do *Squid*. O sistema se resume através de um processo de autenticação e posterior desautenticação através da solução *NoCatAuth*, desenvolvida pela *nocat*. Essa solução permite aliar as duas ferramentas que o *Squid* não permite funcionar ao mesmo tempo: autenticação e *proxy transparente* para os usuários.

Para o administrador, é permitido que apenas usuários autenticados consigam acessar o recurso da *Internet*, além de não necessitar de configurações adicionais no equipamento do usuário final para sua utilização.

## Capítulo 3

# Materiais e Métodos

Nesta Capítulo são apresentados os materiais e métodos utilizados no desenvolvimento deste trabalho, bem como a metodologia e configuração do sistema de autenticação de usuários.

A instituição onde o sistema de autenticação *NoCatAuth* foi implantado, trata-se de um órgão governamental denominado Prefeitura Municipal de Sacramento. A necessidade da implantação do sistema de autenticação, surgiu através da interligação entre todos os setores externos ao prédio municipal, através de uma rede sem fio, com distribuição de acesso à *Internet* controlada a todos os usuários. Por motivos de segurança, a instalação de um ambiente de rede sem fio, necessita de um software que controle a distribuição e funcionamento do acesso à *Internet*, evitando assim que usuários que não façam parte da rede consigam acesso ao serviço, garantindo a integridade do sistema, e da informação.

### 3.1 Materiais

Parte da pesquisa bibliográfica foi realizada em livros específicos que tratam do assunto. Também foram utilizados os recursos da Internet como meio de obtenção de material de pesquisa. Todo o material bibliográfico utilizado para o desenvolvimento deste trabalho está citado e referenciado no texto e nas referências bibliográficas.

Todos os recursos computacionais necessários para o desenvolvimento deste trabalho, foram disponibilizados pela Prefeitura Municipal de Sacramento. Segue as especificações de todos os equipamentos utilizados para a realização deste trabalho:

- 02 computadores com as seguintes configurações



- Processador : Core 2 Duo.
  - Memória: 1024 Ram.
  - HD: 80.0 GB.
  - Placa de Rede: 3Com Ethernet Adapter.
- 4 Switches 16 portas 100Mbps.
  - 170 computadores Clientes.
  - 13 Antenas de acesso a rede sem fio Clientes.
  - 02 Antenas de acesso a rede sem fio para recebimento e passagem do sinal de internet para o provedor local.

## 3.2 Metodologia

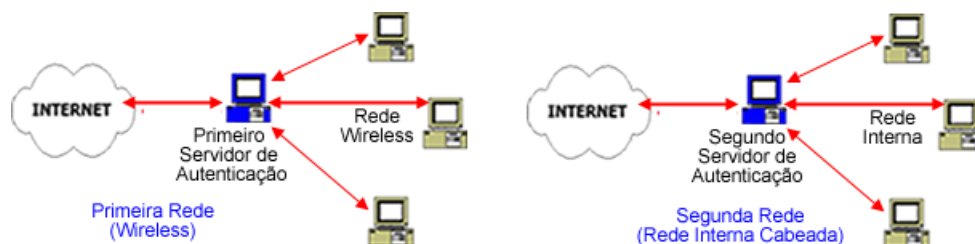
Inicialmente a metodologia aplicada no desenvolvimento deste trabalho consistiu em uma revisão da literatura dos programas necessários para instalação e funcionamento do sistema de autenticação de usuários.

Com base na revisão bibliográfica levantada no decorrer do trabalho, foi possível identificar as principais características que se fazem necessárias para configuração do sistema de autenticação de usuários utilizando de *Proxy transparente*. Desta forma,

O ambiente de implantação do sistema *NoCatAuth* caracteriza-se por um parque de informática contendo 170 equipamentos, interligados em rede através de um servidor com sistema *Linux*, possuindo todos os serviços necessários para seu funcionamento, baseado em *Fedora Core* versão 6.0. O servidor utilizado pelo sistema *NoCatAuth* trabalha unindo duas redes, uma rede externa através do sistema de rede sem fio, onde o parque de informática apresenta-se interligado, e uma rede interna com acesso à *Internet*, onde o responsável em realizar o trabalho de transferência de dados entre o mundo externo e a rede interna é o servidor de *gateway*, através do sistema de autenticação *NoCatAuth* e o sistema de *proxy cache/Squid*.

A implantação iniciou-se em setembro de 2006, através da instalação e configuração do servidor de *gateway* que é responsável em controlar e distribuir o serviço. O primeiro ponto de acesso foi redirecionado no início do mês de outubro, para testes do funcionamento do sistema. O sistema atingiu a sua maturidade em dezembro de 2006, possuindo um parque de 170 máquinas interligadas, através de 13 pontos distintos. O sistema de autenticação atual, conta com dois servidores, localizados no prédio principal da instituição. O primeiro servidor de autenticação, interliga 13 pontos de acesso de rede à *Internet*, possuindo um parque de 100

equipamentos. O segundo servidor de autenticação, interliga os equipamentos do prédio principal da instituição à *Internet*, possuindo um parque de 70 equipamentos. A necessidade de dois servidores de *gateway*, surgiu apartir da solicitação de redes distintas, garantindo a segurança dos setores internos, por se tratarem de órgãos sem dependência, conforme pode-se observar na figura 3.1 .

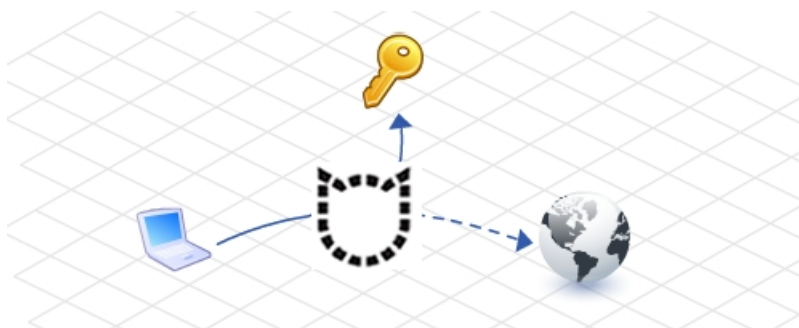


**Figura 3.1:** Situação atual do sistema do autenticação da instituição

### 3.3 Instalações e Configurações

#### 3.3.1 O sistema *NoCatAuth*

O *software NoCatAuth gateway* funciona baseado no conceito de *gateway de captura*. Nessa técnica, um *host* cliente localizado em uma rede atrás do *NoCatAuth* necessita se autenticar para obter acesso a *Internet*, conforme observado na figura 3.2.



**Figura 3.2:** Captura de tráfego pelo sistema *NoCatAuth gateway* (BRANDÃO, 2004)

Como existem muitas tecnologias para prover autenticação e cada uma exige um tipo de infra-estrutura ou softwares diferentes, a técnica de captura resume-se apenas em exibir para o usuário uma tela de login no navegador. Segundo

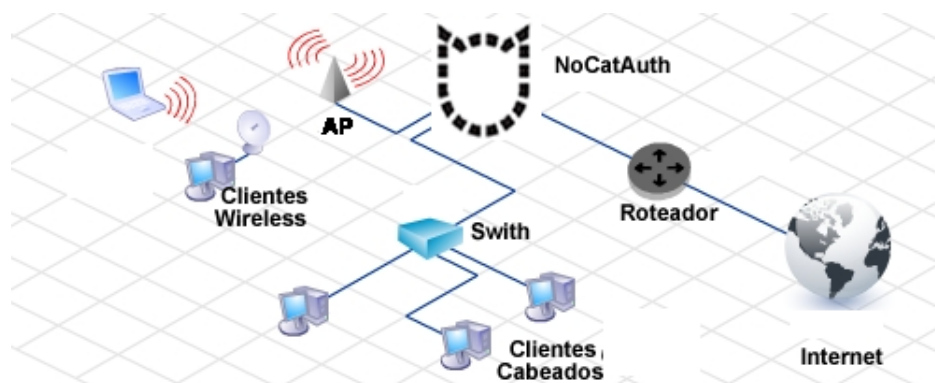
(BRANDÃO, 2004), o software de autenticação *NoCatAuth*, desenvolvido pela *Nocat*, escrito em *Perl*, captura o tráfego da porta 80 e o desvia para o servidor *HTTP* seguro, onde uma página solicitando nome e senha e uma tela para cadastro de novos usuários. A tela de login e registro de novos usuários são compostas por uma página *HTML* contendo *scripts* em *Perl*, com acesso ao banco de dados *MySQL*.

O *NoCatAuth* não utiliza nenhuma outra tecnologia de autenticação. A base de dados de usuários fica armazenada em um banco de dados *MySQL* local.

Para que o usuário possa ser redirecionado até página de *login* do sistema de autenticação, é necessário que a infra-estrutura da rede esteja totalmente funcional (meio físico cabeado ou *rede sem fio*, etc...). Nesse ponto, o *host* cliente deve estar disponível para comunicações *TCP-IP*.

Com a conexão de rede funcional, o *host* cliente necessitará apenas que o Navegador reconheça o protocolo *HTTP* 1.0 ou superior (*Internet Explorer*, *Firefox*, etc...)

O *gateway* que proverá o recurso de autenticação deve estar no caminho direto de acesso para a *Internet*, ou seja, todo tráfego deverá passar pelo *NoCatAuth*, conforme pode-se observar na figura 3.3. Somente assim, o *NoCatAuth* poderá diferenciar acessos autorizados, que já possuem permissões e não autorizados, que deverão ser autenticados para trafegar na rede. Uma vez que um *host* cliente se autenticou, as permissões para trafegar na rede são concedidas.

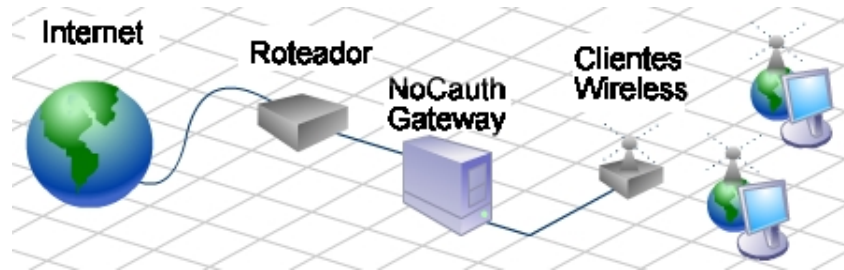


**Figura 3.3:** Estrutura física de rede para o funcionamento correto do sistema *NoCatAuth gateway* (BRANDÃO, 2004)

O *NoCatAuth* não possui controle sobre os meios físicos a que está conectado, ou seja, ele não provê recursos de enlace de dados como *Access Point*, *VLAN*, etc... Suas funções são específicas para portal de autenticação.

A topologia da rede onde o *NoCatAuth* será instalado pode ser variada e heterogênea, bastando apenas atender aos requisitos básicos para instalação do sistema,

conforme pode-se observar na figura 3.4 . O *NoCatAuth* apenas verifica no *firewall* do *Linux* se a máquina em questão possui o *IP* liberado para navegação, tomando a decisão de autenticação do usuário ou apenas resposta da requisição solicitada.



**Figura 3.4:** Características do sistema NoCatAuth gateway (BRANDAO, 2004)

O *NoCatAuth* pode ser configurado para trabalhar em três modos:

- *Open* - O tráfego na porta 80 é capturado e redirecionado para uma página de splash, que após ter sido exibida libera o tráfego de *Internet*.
- *Passive* - O tráfego dos usuários não autenticados para a porta 80 será capturado, onde após autenticação, uma regra de *firewall* (*iptables*) permitirá o acesso à *Internet* por meio de *NAT*.
- *Captive* - Semelhante ao *Passive*, porém, ao se identificar com sucesso, o acesso a *internet* será liberado, mas sem *Nat*.

O *NoCatAuth* funciona com duas instâncias nos modos *PASSIVE* e *CAPTIVE*, uma, *NoCatAuth gateway*, e outra *NoCatAuth Auth*, onde o *gateway* redirecionará o tráfego *HTTP* para a porta 80 do servidor de autenticação, que após autenticar o usuário, se conectará a porta 5280 do *gateway* para dar permissão de passagem ao *Host*. A conexão do servidor *NoCatAuth Auth* na porta 5280 do servidor *No-CatAuth gateway* é criptografada com *PGP* para dar suporte a casos em que o servidor *NoCatAuth Auth* não possua uma conexão confiável com o servidor *No-CatAuth gateway*.

O sistema *NoCatAuth* possui as seguintes características:

- Acesso à *Internet* liberado somente a usuários autenticados.
- Autenticação em uma página *web* usando banco de dados *MySQL*.
- Senha de autenticação de usuários armazenada em banco de dados *MySQL* usando de criptografia *GnuPg*.

- Uso de conexão segura através do servidor de *web Apache* na autenticação.
- Conexão dos usuários à *Internet* através de *NAT*, sendo realizado pelo serviço *NoCatAuth gateway*.
- Instalação e configuração dos serviços de *gateway* e autenticação no mesmo equipamento.

### 3.3.2 Pré-requisitos para instalação do sistema *NoCatAuth*

O *NoCatAuth* é um software que necessita trabalhar sob condições aceitáveis para prover estabilidade e bom funcionamento. Algumas condições e pré-requisitos devem ser observados:

**Tabela 3.1:** Pré-requisitos do *Sistema*

Hardware	Processador Memória RAM Disco Rígido Interfaces de rede	1.0 Ghz (32 bits) 256 Mb 40 Gb duas
Sistema Operacional	Distribuição Linux: Tipo de instalação	Slackware 11.0 ou Fedora Core 6.0 Completa
Softwares Envolvidos	<i>Kernel</i> Servidor <i>web</i> Chaves criptografadas Sistema de Autenticação Banco de dados Linguagem de programação Fornecimento de <i>IP</i> da rede Servidor de nomes Firewall	<i>Linux</i> 2.4.26 ou superior Apache 2.0 usando módulo <i>SSL</i> <i>GnuPg</i> 1.2.2 <i>NoCauth</i> 0.82 <i>MySQL</i> 4.0 ou superior <i>Perl</i> 5.8.4 <i>Dhcpd</i> 1.3.0 <i>Bind</i> 9.2.3 <i>IPtables</i> 1.2.10

### 3.3.3 Instalação dos Módulos Adicionais *Perl*

O sistema de *gateway* e autenticação *NoCatAuth* necessitam de módulos adicionais da linguagem *Perl* para seu funcionamento. A linguagem *Perl* pode ser instalada através do comando “*yum -y perl\**” para distribuições baseadas em *Fedora Core* ou “*apt-get perl\**” para instalações baseadas no *Debian*, podendo ainda ser selecionado manualmente na instalação inicial do sistema.

Os módulos *Perl* usados pelo sistema *NoCautAuth* são:

- *Net::Netmask*
- *DBI*
- *DBD::mysql*
- *Digest::MD5*

Os módulos adicionais da linguagem *Perl* podem ser instalados através do módulo *CPAN*, ou através de pacotes fornecidos para cada distribuição. A primeira solicitação ao módulo *CPAN* da linguagem *Perl*, através do comando em *shell linux* “*perl -MCPAN -e shell*” serão solicitadas informações como diretório de instalação e configurações do sistema, onde o administrador poderá usar o padrão de instalação, pressionando a tecla *N* para todas as opções.

Os módulos adicionais devem ser instalados posteriormente com o comando *install*, conforme pode-se observar na figura 3.5. O módulo *DBD::mysql*, em algumas distribuições como o *Fedora Core*, apresenta erros na compilação, sendo necessário a utilização da cláusula *force* antes do comando *install*. O primeiro módulo que necessita ser atualizado anterior a instalação dos demais módulos é a própria *CPAN* do *perl*, através do comando interno a *CPAN* “*install Bundle::CPAN*”, que garante a instalação atual de todos os módulos assim que solicitados e da própria linguagem *Perl*.

```
perl -MCPAN -e shell
cpan> install Bundle::CPAN
cpan> install DBI
cpan> install DBD::mysql
cpan> install Digest::MD5
cpan> install Net::Netmask
cpan> exit
```

**Figura 3.5:** Instalação dos módulos adicionais e da biblioteca CPAN (BRANDAO, 2004)

A verificação da instalação dos módulos adicionais da linguagem *Perl* pode ser efetuada através da chamada ao nome da linguagem seguida do módulo a verificar, conforme pode-se observar na figura 3.6. O módulo instalado não poderá retornar nenhuma mensagem de erro quando invocado pela linguagem, que na realidade é a garantia de instalação bem sucedida do módulo.

```
perl -MDBI
perl -MDBD:mysql
perl -MDigest::MD5
perl -MNet::Netmask
```

**Figura 3.6:** Verificação dos módulos instalados (BRANDAO, 2004)

### 3.3.4 Instalação do Sistema

Todos os pacotes necessários para instalação do sistema de autenticação e gateway *NoCatAuth* podem ser obtidos no site do desenvolvedor <sup>1</sup>, via *download*, conforme pode-se observar na figura 3.7, ou através do comando de texto *wget* usado pelo *Linux*, conforme pode-se observar na figura 3.8. Após a obtenção do sistema, o administrador descompactará o arquivo de instalação em um diretório qualquer para iniciar os procedimentos de instalação. (BRANDAO, 2004)

```
wget http://nocat.net/download/NoCatAuth/NoCatAuth-0.82.tar.gz

tar -xvzf NoCatAuth-0.82.tar.gz

cd NoCatAuth-0.82
```

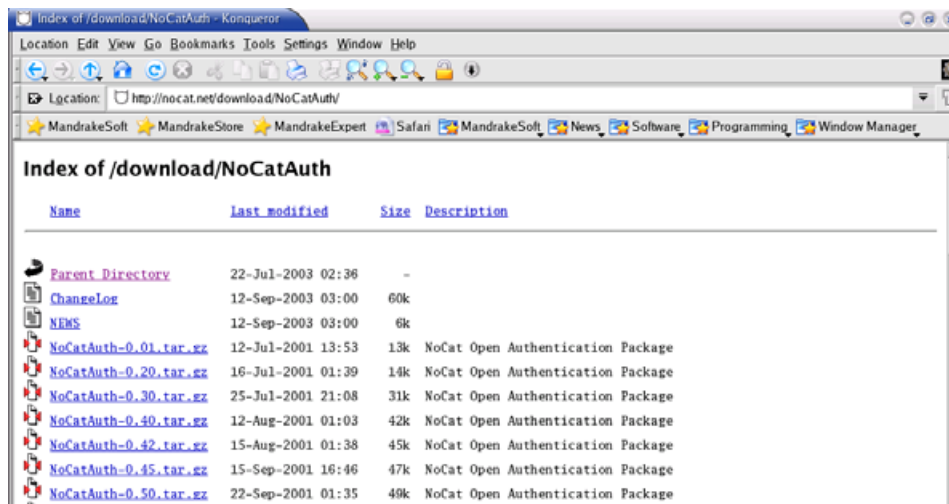
**Figura 3.7:** Obtenção via *wget* e extração dos arquivos de configuração (BRANDAO, 2004)

O sistema *NoCatAuth*, na sua concepção inicial, foi desenvolvido para o funcionamento em sistemas *Linux* com *firewall iptables* na versão 2.4. Para a instalação correta em sistemas baseados em *firewall iptables* na versão 2.6, torna-se necessário a modificação do arquivo de detecção de *firewall*, interno ao sistema. O arquivo de detecção de *firewall*, encontra-se localizado no diretório *NoCatAuth-0.82/bin/* com o nome *detect-fw.sh*, que pode ser editado com um editor nativo do *Linux* denominado *vi*, para posterior compilação do sistema,, conforme pode-se observar na figura 3.9.

A instalação do sistema de autenticação e *gateway NoCatAuth*, pode ser realizada com a execução do comando *make*. O comando *make* deve ser executado interno ao diretório principal do sistema *NoCatAuth* *NoCatAuth-0.82/*, conforme pode-se observar na figura 3.10. Após a execução da instalação, o sistema de autenticação *NoCatAuth*, estará disponível para configuração no diretório de sistema */usr/local/nocat*.

---

<sup>1</sup><http://nocat.net/download/NoCatAuth/NoCatAuth-0.82.tar.gz>



**Figura 3.8:** Obtenção do arquivo de instalação via *download*

```
export PATH=$PATH:/sbin:/usr/sbin:/usr/local/sbin
# Have we been explicitly told which firewall
#scripts to install?

if [ -n "$1" -a -n "$2" -a -d "$2/$1" ]; then
    FIREWALL=$1
    shift

##Alteração para firewall 2.6 do sistema Fedora Core 6.0
##Padrão 2.6
# Do we have iptables *and* are running Linux 2.6?
#
elif which iptables >/dev/null 2>&1 && \
    test X`uname -sr | cut -d. -f-2` = X"Linux 2.6"; then
    FIREWALL=iptables
    FW_BIN=iptables

#
```

**Figura 3.9:** Alterações arquivo detect-fw.sh para o funcionamento *Firewall 2.6* - sistema *Fedora Core 6.0*



```
make PREFIX=/usr/local/nocat/gw gateway  
  
make PREFIX=/usr/local/nocat/authserv authserv
```

**Figura 3.10:** Instalação do sistema *NoCatAuth* no modo *gateway* e autenticação (BRANDAO, 2004)

### 3.3.5 Criação de chaves criptografadas em *GnuPg*

As chaves criptografadas geradas através de *GnuPg* garantem, com o uso de criptografia a segurança da comunicação de dados entre o cliente e o banco de dados *MySQL* na autenticação. A chave *GnuPg* deve ser a mesma para ambas instâncias, para que elas se comuniquem, portanto, ela será criada no servidor de autenticação, sendo posteriormente copiada no servidor de *gateway*.

A criação de chaves envolvem várias perguntas respondidas pelo administrador do sistema, onde a senha da chave não deve ser fornecida, ficando assim vazia, para possibilitar a comunicação entre o banco de dados e cliente solicitante. Esta chave gerada possibilita uma segurança maior ao banco de dados *MySQL*, criptografando as senhas utilizadas.

A criação de chaves é realizada através do comando do *Linux* “*make*”, recebendo como parâmetros o diretório de instalação do sistema *NoCatAuth* que contém a chave. O processo de criação de chaves criptografadas pode ser observado na figura 3.11 e na figura 3.12. As chaves criptografadas geradas através do sistema *GnuPg*, devem permitir ao sistema *NoCatAuth* a leitura e execução, possibilitando assim, através da criptografia, criar a senha armazenada no banco de dados *MySQL*.

```
# make PREFIX=/usr/local/nocat/authserv/pgp pgpkey  
# cd /usr/local/nocat/authserv/pgp  
# cp trustedkeys.gpg /usr/local/nocat/gw/pgp
```

**Figura 3.11:** Criação de chaves criptografadas em *GnuPg* (BRANDAO, 2004)

### 3.3.6 Criação Certificados Digitais usados pelo servidor *web Apache*

Os certificados digitais gerados para o servidor *web Apache*, garantem a comunicação criptografada entre o servidor *web Apache* e o cliente, evitando que algum programa *sniffer* obtenha o nome do usuário e senha de um dos usuários cadastrados no sistema. Os certificados digitais usados para garantir a segurança no servidor *web Apache*, podem ser gerados na instalação do sistema, ou podem ser

```

Por favor selecione o tipo de chave desejado:
  (1) DSA and Elgamal (default)
  (2) DSA (apenas assinatura)
  (5) RSA (apenas assinatura)
Sua opção? 1
What keysize do you want? (2048) 1024
O tamanho de chave pedido é 1024 bits
Por favor especifique por quanto tempo a chave deve ser válida.
  0 = chave não expira
  <n> = chave expira em n dias
  <n>w = chave expira em n semanas
  <n>m = chave expira em n meses
  <n>y = chave expira em n anos
A chave é valida por? (0) 0
Key does not expire at all
Is this correct? (y/N) y

Nome completo: Carlos Humberto Ribeiro
Endereço de correio eletrônico: pmsinfo@sacranet.com.br
Comentário:
Você selecionou este identificador de usuário:
  "Carlos Humberto Ribeiro <pmsinfo@sacranet.com.br>"
Muda (N)ome, (C)omentário, (E)ndereço ou (O)k/(S)air? 0

```

**Figura 3.12:** Processo de criação de chaves criptografadas com *GnuPg* com recebimento de dados

obtidos por terceiros, através de entidades certificadoras como *VeriSign*, mediante pagamento.

Os certificados digitais gerados pelo servidor *web Apache*, por padrão são gerados no diretório *ssl*, dentro do diretório de instalação do servidor *web Apache*, que posteriormente será referenciado dentro do arquivo de configuração específico, conforme pode-se observar na figura 3.13. Este padrão é adotado para garantir a facilidade na manipulação dos arquivos *Linux* relacionados a cada programa. A distribuição denominada *Fedora Core* por padrão instala os arquivos de configuração do servidor *web Apache* no diretório */etc/httpd*.

```

# cd /etc/httpd,
# mkdir ssl
# cd ssl

```

**Figura 3.13:** Criação do diretório para armazenamento de certificados digitais (BRANDAO, 2004)

Os certificados digitais gerados para o servidor *web Apache* devem permitir a leitura e execução, bem como pertencerem ao usuário *apache* e grupo *apache*, como realizado nos sistemas de chaves criptografadas no sistema *GnuPg*.

A criação dos certificados digitais gerados para o servidor *web Apache* com criptografia *GnuPg* pode ser realizada através do comando *Linux* “*openssl*”, conforme pode-se observar na figura 3.14.

```
# openssl genrsa -out nocat.key 1024
# openssl req -new -key nocat.key -out nocat.csr
# openssl x509 -days 365 -req -in nocat.csr -signkey nocat.key
  -out nocat.crt
```

**Figura 3.14:** Geração dos certificados digitais (BRANDAO, 2004)

### 3.3.7 Configuração do servidor *web Apache* e do certificado digital

O servidor *web apache* é configurado em todas as suas distribuições através do arquivo `httpd.conf`, no diretório de instalação do servidor *web Apache*, de acordo com os padrões de cada distribuição. Para que o sistema *NoCatAuth* funcione corretamente o servidor *web Apache* deve permitir a execução de *Cgi*, além de referenciar a mesma para o diretório padrão de instalação do sistema *NoCatAuth*, conforme pode-se observar na figura 3.15.

```
ScriptAlias /cgi-bin/ "/var/www/nocat/authserv/cgi-bin/"
<Directory /var/www/nocat/authserv/cgi-bin/>

    SetEnv PERL5LIB /var/www/nocat/authserv/lib

    SetEnv NOCAT /var/www/nocat/authserv/nocat.conf

</Directory>
```

**Figura 3.15:** Configuração dos parâmetros para *CGI-bin* no diretório de instalação *NoCatAuth* (BRANDAO, 2004)

Além das alterações para execução *CGI* (*Common Gateway Interface*) no diretório de instalação do sistema *NoCatAuth*, o servidor *web Apache* deve apresentar as configurações básicas de um servidor *web*, sendo opcional o redirecionamento do diretório raiz para a porta 5280, onde o sistema *NoCatAuth* trabalha, conforme pode-se observar na figura 3.16.

```

NameVirtualHost 10.10.12.254:80

<VirtualHost 10.10.12.254:80>

    ServerAdmin pmsinfo@sacranet.com.br

    DocumentRoot /var/www/html

    ServerName localhost

    ErrorLog logs/localhost-error_log

    CustomLog logs/localhost-access_log common

    Redirect Permanent / http://10.10.12.254:5280

</VirtualHost>

```

**Figura 3.16:** Configuração do diretório de execução do servidor *web Apache* com redirecionamento para a porta padrão do sistema *NoCatAuth*

Para que o certificado digital gerado para o servidor *web Apache* possa ser utilizado e realizar a criptografia de comunicação entre o servidor e o cliente, é necessário a configuração do arquivo `ssl.conf`, localizado no diretório de instalação dos arquivos de configuração servidor *web*, no subdiretório `ssl`.

O arquivo `ssl` deve referenciar o diretório onde se encontra a chave, para que a mesma possa ser lida na execução segura do servidor através da chamada no *Browser*, conforme pode-se observar na figura 3.17.

### 3.3.8 Configuração dos parâmetros do sistema *NoCatAuth*

O sistema *NoCatAuth* é baseado em dois arquivos de configuração, sendo que o primeiro se referencia ao sistema de autenticação e o segundo ao sistema de *gateway*. A configuração é realizada em dois arquivos em separado, para evitar que o sistema de autenticação e o sistema de *gateway* necessitem ficar instalados no mesmo equipamento.

Após a instalação do sistema *NoCatAuth* usando o mesmo equipamento, serão criados automaticamente os diretórios `/usr/local/nocat/gw` para o servidor de *gateway* e `/usr/local/nocat/authserv` para o servidor de autenticação, contendo internamente o arquivo `nocat.conf` que trata-se do arquivo de configuração

```

LoadModule ssl_module modules/mod_ssl.so
Listen 443
AddType application/x-x509-ca-cert .crt
AddType application/x-pkcs7-crl .crl
SSLPassPhraseDialog builtin
SSLSessionCache dbm:/etc/httpd/logs/ssl_scache
SSLSessionCacheTimeout 300
#SSLMutex file:/etc/httpd/logs/ssl_mutex
SSLMutex default
SSLRandomSeed startup builtin
SSLRandomSeed connect builtin
SSLCryptoDevice builtin
<VirtualHost _default_:443>
DocumentRoot /var/www/nocat/authserv/htdocs
ServerName *:443
ServerAdmin pmsinfo@sacranet.com.br
ErrorLog /etc/httpd/logs/erros_log
TransferLog /etc/httpd/logs/access_log
LogLevel warn
SSLEngine on
SSLProtocol all -SSLv2
SSLCipherSuite ALL:!ADH:!EXPORT:!SSLv2:RC4+RSA:+HIGH:
+MEDIUM:+LOW
SSLCertificateFile /etc/httpd/ssl/nocat.crt
SSLCertificateKeyFile /etc/httpd/ssl/nocat.key
<Files ~ \.(cgi|shtml|phtml|php3?)$>
    SSLOptions +StdEnvVars
</Files>
<Directory /var/www/nocat/authserv/cgi-bin>
    SSLOptions +StdEnvVars
</Directory>
SetEnvIf User-Agent .*MSIE.* \
    nokeepalive ssl-unclean-shutdown \
    downgrade-1.0 force-response-1.0
CustomLog logs/ssl_request_log \
    "%t %h %{SSL_PROTOCOL}x %{SSL_CIPHER}x \"%r\" %b"
</Virtualhost>

```

**Figura 3.17:** Modificação no arquivo de configuração *ssl.conf* para a leitura da chave de certificação digital do servidor *web Apache*

do sistema, bem como as telas utilizadas pelo servidor *web Apache* para realizar o serviço de autenticação e cadastro de usuários.

A configuração do sistema de autenticação baseia-se no arquivo *nocat.conf* interno ao diretório de instalação do servidor, conforme pode-se observar na figura

3.18, sendo obrigatório para o funcionamento a configuração dos seguintes parâmetros:

```
Verbosity          10
PGPKeyPath         /var/www/nocat/authserv/pgp
HomePage           http://www.sacramento.mg.gov.br/
DocumentRoot       /var/www/nocat/authserv/htdocs
DataSource          DBI
Database           dbi:mysql:database=nocat
DB_User            nocat
DB_Passwd          Carlos1001
UserTable          member
UserIDField        login
UserPasswdField    pass
UserAuthField      status
UserStampField     created
GroupTable         network
GroupIDField       network
GroupAdminField    admin
Localgateway       200.243.249.201
LocalNetwork       200.243.249.192/255.255.255.192
LoginForm          login.html
LoginOKForm        login_ok.html
FatalForm          fatal.html
ExpiredForm        expired.html
RenewForm          renew.html
PassiveRenewForm   renew_pasv.html
RegisterForm       register.html
RegisterOKForm     register_ok.html
RegisterFields     name url description
UpdateForm         update.html
UpdateFields       url description
```

**Figura 3.18:** Configuração dos parâmetros usados pelo servidor de autenticação internos ao arquivo *nocat.conf*

- Diretório da chave *GnuPg*, para criação de senhas criptografadas no banco de dados *MySQL*.
- Página do servidor de *gateway* para redirecionamento posterior a autenticação.
- Diretório dos arquivos lidos pelo servidor *web Apache* que realizarão a autenticação do usuário, entre outras telas exibidas para o usuário final no processo de autenticação.

- Tipo de banco de dados usado na autenticação.
- Usuário e senha do banco de dados *MySQL*.
- Endereço de *IP* do *gateway*, e da rede local usado pelo servidor de autenticação.
- Configuração dos nomes dos formulários usados pelo servidor *web Apache*, na solicitação dos serviços, localizados no diretório configurado anteriormente para a leitura do servidor *web Apache*.

A configuração do sistema de *gateway* baseia-se no arquivo *nocat.conf* interno ao diretório de instalação do servidor de *gateway*, conforme pode-se observar na figura 3.19, sendo obrigatório para o funcionamento a configuração dos seguintes parâmetros:

```

Verbosity          10
gatewayName        gateway Prefeitura Sacramento
gatewayMode        Passive
gatewayLog         /etc/httpd/logs/nocat.log
LoginTimeout       86400
StatusForm         status.html
TrustedGroups      Any
Owners             pmsinfo@sacranet.com.br
AuthServiceAddr    10.10.12.254
AuthServiceURL     https://$AuthServiceAddr/cgi-bin/login
LogoutURL          https://$AuthServiceAddr/logout.html
ExternalDevice     eth0
InternalDevice     eth1
LocalNetwork       10.10.12.0/24
DNSAddr            200.243.249.132 200.243.249.134 200.243.249.130
IgnoreMAC          1
IncludePorts       22 80 443
ExcludePorts       25 110
gatewayPort        5280

```

**Figura 3.19:** Configuração dos parâmetros usados pelo servidor de *gateway* internos ao arquivo *nocat.conf*

- Nome do *gateway*.
- Modo de funcionamento do *gateway*.
- Local e nome do arquivo do log usado pelo servidor.

- Tempo de expiração do login, a partir do qual o usuário será forçado a se reconectar ao sistema, sendo configurado em segundos.
- Página de redirecionamento onde encontra-se o servidor de autenticação.
- Diretório dos arquivos lidos pelo servidor *web Apache* no servidor de *gateway*, usados para telas de *splash* e de *status* do servidor. Ambos os servidores possuem todos os modelos de telas usadas pelo servidor *gateway*, sendo que o servidor de autenticação utiliza as telas relacionadas ao serviço de autenticação, e o servidor de *gateway* utiliza as telas relacionadas ao serviço de *gateway*.
- Nome da tela de *Splash*
- Nome da tela de *Status*
- Email do administrador do sistema
- Endereço do servidor de autenticação.
- Endereço da *URL* de *login*
- Endereço da *URL* de *logout*
- Interface utilizada para rede externa de *internet* usada com *IP* válido.
- Interface utilizada para rede interna, de comunicação com os clientes, com *IP* inválido.
- Definição da classe e rede interna.
- Dns utilizados para conexão a *Internet*
- Parâmetros de endereço de *MAC* para *Bridge*.
- Portas a serem incluídas na captura além da porta padrão 80.
- Portas excluídas de autenticação, como por exemplo a porta 25 e 110, poderiam ser excluídas, permitindo assim o usuário a leitura de *Emails* sem autenticação prévia.
- Porta padrão de funcionamento do servidor de *gateway*.



### 3.3.9 Importação da Tabela de nome *nocat* do Banco de dados *MySQL*

O Sistema *NoCatAuth*, gerencia seus usuários através de um banco de dados em *MySQL*, que no momento da instalação do sistema, não apresenta disponível para acesso. O banco de dados deve ser criado sobre o nome *nocat* concedendo todos os privilégios de trabalho sobre ele, conforme pode-se observar na figura 3.20.

```
# mysql -u root -psenhadorootmysqlaqui
mysql> CREATE DATABASE nocat;
mysql> GRANT ALL PRIVILEGES ON nocat.* TO nocat@localhost
      IDENTIFIED BY "senhadonocat" WITH GRANT OPTION;
mysql> FLUSH PRIVILEGES;
mysql> exit
```

**Figura 3.20:** Criação do banco de dados *MySQL* denominada *nocat* responsável pelo armazenamento dos usuários do sistema

A disponibilização do banco de dados para uso, envolve duas etapas, sendo que a primeira realiza a criação do banco e liberação dos poderes de manipulação do mesmo, e a segunda realiza a importação das tabelas em branco usadas pelo sistema *NoCatAuth*, através de um arquivo de script presente na instalação do sistema, conforme pode-se observar na figura 3.21.

```
mysql -u root -psenhadorootmysqlaqui nocat <
  <Diretorio_Sistema>/NoCatAuth-0.82/etc/nocat.schema
```

**Figura 3.21:** Importação da tabela de dados *MySQL* denominada *nocat* responsável pelo armazenamento dos usuários do sistema (BRANDAO, 2004)

### 3.3.10 Ajustes Finais de atributos relacionados aos arquivos

Para o funcionamento correto do sistema de autenticação *NoCatAuth* são necessários alguns ajustes em permissões de arquivos, permitindo que o sistema *Linux* consiga, por exemplo, ler os arquivos de chaves *GnuPg* para criação das chaves

criptografadas dentro do banco de dados, além da permissão para leitura ao diretório do servidor de autenticação ao grupo do servidor *web Apache*, permitindo assim que os arquivos sejam executados e mostrados, não retornando assim nenhuma mensagem de erro.

A primeira alteração refere-se a permissão ao diretório de chaves *GnuPg*. Para que a leitura da chave seja possível e processada dentro do sistema, o servidor *web Apache* necessita ser dono dos arquivos de chaves, através da execução do comando *Linux* “*chown apache:apache \* -R*”, que é responsável para indicar para o sistema *Linux* que o dono e o grupo dos arquivos mencionados passa a ser o usuário do servidor *web Apache*. Além da alteração do dono do arquivo é necessária a mudança das permissões do arquivo, permitindo a leitura e execução do grupo de usuários do servidor *web Apache*, bem como os outros usuários do sistema, através do comando *Linux* “*chmod 755 \**”, executando no diretório de criação das chaves criptografadas, tanto para o diretório de chaves do servidor de *gateway*, como para o diretório do servidor de autenticação.

A segunda alteração se refere a mudança do dono e do grupo do diretório de autenticação do sistema *NoCatAuth* para o usuário do servidor *web Apache*, mudando também a permissão de leitura dos arquivos apenas para o usuário do servidor *web Apache*, garantindo assim a integridade do sistema de autenticação. Através do comando *Linux*: “*chown apache:apache /usr/local/nocat/authserv -R*”, o novo dono e grupo do arquivos relacionados no diretório *authserv* passa a ser o usuário do sistema *Linux, Apache*. A mudança das permissões de acesso aos arquivos é realizada através do comando *Linux* “*chmod 700 /usr/local/nocat/authserv -R*”, permitindo assim a leitura e execução dos arquivos contidos neste diretório, apenas pelo servidor *web Apache*.

### **3.3.11 Configuração do sistema de filtro de pacotes *Iptables***

Para o funcionamento do sistema de autenticação e *gateway NoCatAuth*, o filtro de pacotes *Iptables* fica responsável em redirecionar todos os pacotes recebidos, para a porta padrão de funcionamento do *NoCatAuth*, a porta 5280. O redirecionamento dos pacotes, garantem o funcionamento do sistema, solicitando ou não a autenticação do usuário assim que o pacote chega a porta 5280.

Em sistemas de distribuição de acesso a *Internet* baseados em um *cache/Squid*, na concepção simples, todos os pacotes que atravessam o *firewall* do servidor devem ser redirecionados para o sistema de cache para a resolução da *URL* solicitada, respondendo assim a requisição do usuário. A diferença entre o *Squid* e o servidor *NoCatAuth*, é que o *Squid* redireciona o tráfego para a porta padrão de funcionamento do servidor, normalmente a porta 3128, conforme pode-se observar na figura 3.22.

```
iptables -t nat -A PREROUTING -s 192.168.0.0/24 -p tcp --dport
      80 -j REDIRECT --to-port 3128

iptables -t nat -A POSTROUTING -s 192.186.0.0/24 -j MASQUERADE
```

**Figura 3.22:** Redirecionamento das requisições realizadas no servidor de *proxy*, *Squid*

O servidor de autenticação redireciona o tráfego para a porta padrão do sistema *NoCatAuth*, a porta 5280, onde será realizada ou não a autenticação do usuário, sendo posteriormente passado ao *Squid* para a navegação, conforme pode-se observar na figura 3.23. Assim que o usuário é autenticado, o sistema *NoCatAuth*, apenas redireciona o pacote para o *Squid*. Caso o usuário não possua autenticação, é solicitado que o mesmo se autentique, para que o sistema *NoCatAuth*, redirecione suas requisições para o *Squid*.

```
iptables -t nat -A PREROUTING -s 192.168.0.0/255.255.0.0 -d
! 10.10.12.254 -p tcp -m multiport --dports
21,22,80,563,2021,70,210,280,488,591,777,1863,5190 -j DNAT
--to-destination 200.243.249.201:3128

iptables -t nat -A PREROUTING -s 192.168.0.0/255.255.0.0 -d !
10.10.12.254 -p udp -m multiport --dports
21,22,80,563,2021,70,210,280,488,591,777,1863,5190 -j DNAT
--to-destination 200.243.249.201:3128
```

**Figura 3.23:** Redirecionamento padrão para requisições realizadas no sistema de autenticação *NoCatAuth*

### 3.3.12 Arquivo de inicialização do sistema *NoCatAuth*

Após configurados os parâmetros dos arquivos do sistema *NoCatAuth*, torna-se necessário a criação do arquivo de inicialização que será adicionado ao sistema para carregar o *NoCatAuth* na inicialização do sistema *Linux*, conforme pode-se observar na figura 3.24. Este arquivo poderá ser chamado através do arquivo `/etc/rc.d/rc.local` em sistemas baseados na distribuição *Slackware*, ou adicionado através do comando “`chkconfig –add <nome do arquivo executável>`” em distribuições baseadas em *Fedora Core*. O arquivo de texto usado para inicialização do sistema, deve permitir execução e leitura sendo previamente configurado com o comando `chmod 755` do *Linux*, para posterior execução.

```

. /etc/rc.d/init.d/functions
. /etc/sysconfig/network
[ ${NETWORKING} = "no" ] && exit 0
[ -f /var/www/nocat/gw/nocat.conf ] || exit 0
RETVAL=0
prog="nocat gateway"
start() { echo -n $"Starting $prog: "
    daemon /var/www/nocat/gw/bin/gateway
    RETVAL=$?
    echo [ $RETVAL -eq 0 ] && touch /var/lock/subsys/nocat
    return $RETVAL }
stop() {echo -n $"Shutting down $prog: "
    killproc gateway
    RETVAL=$?
    echo [ $RETVAL -eq 0 ] && rm -f /var/lock/subsys/nocat
    return $RETVAL}
case "$1" in
start) start ;;
stop) stop ;;
restart|reload) stop
    start
    RETVAL=$?;;
condrestart)if [ -f /var/lock/subsys/nocat ]; then
    stop
    start
    RETVAL=$?\
    fi ;;
status)status nocat
    RETVAL=$? ;;
*) echo $"Usage: $0 {start|stop|restart|condrestart|status}"
    exit 1
esac exit $RETVAL

```

**Figura 3.24:** Arquivo de Inicialização do sistema *NoCatAuth*

### 3.3.13 Considerações Finais

A solução oferecida pelo *NoCatAuth* permite a utilização de um *proxy* transparente com autenticação do usuário. Entretanto, nota-se que o sistema de autenticação necessita de algumas implementações para tornar-se um sistema funcional. No Capítulo 4 é modelada uma proposta para melhoria das telas de capturas de senhas, além de mudanças na tela principal do sistema, protegendo ao acesso ao registro de novos usuário, através de uma tela de autenticação para o usuário administrador.

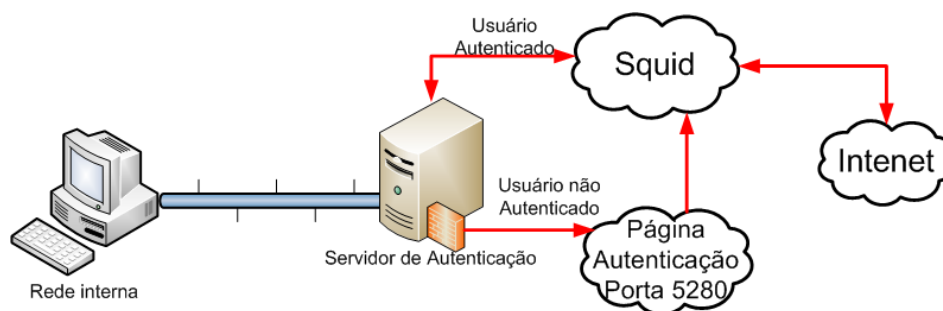


## Capítulo 4

# Funcionamento do sistema de autenticação *NoCatAuth*

Para que o sistema *NoCatAuth* atinja a maturidade no campo comercial ou obtenha o potencial de um sistema de autenticação, torna-se necessário a realização de algumas modificações nas telas que o sistema *NoCatAuth* apresenta para o usuário final, além da separação de alguns itens importantes, apenas de responsabilidade do administrador do sistema.

O sistema de autenticação *NoCatAuth*, por padrão responde a requisições do sistema, na porta 5280, conforme pode-se observar na figura 4.1. Assim que uma conexão é realizada por um cliente qualquer, possuindo uma solicitação de um endereço, o sistema *NoCatAuth* solicita do *firewall* a confirmação se o cliente possui autorização para navegação, tomando a decisão de autenticação ou liberação do conteúdo da página solicitada, através do *Squid*.



**Figura 4.1:** Funcionamento do Sistema de Autenticação *NoCatAuth*

O processo de autenticação, quando solicitado pelo sistema, resume-se em exibir uma página em formato *HTML*, solicitando do cliente, o *login* e senha de usuário previamente cadastrado. A liberação do acesso à *Internet* é realizado após a confirmação do usuário e senha em um banco de dados *MySQL*,textitInternet se concretiza com a exibição de uma tela de boas-vindas ao sistema, contendo na parte superior esquerda uma tela de *Splash*, com informações sobre o tempo de expiração do usuário no sistema. O tempo de expiração é exibido em segundos e configurado previamente na instalação do sistema de autenticação.

A tela de *Splash*, além de informações sobre o tempo de permanência do usuário no sistema, apresenta um botão rotulado como *logout*, que pode ser utilizado para desconexão do cliente assim que o recurso não seja mais necessário. Esta tela de *Splash* pode ser minimizada para permitir a utilização da *Internet*, onde posteriormente será restaurada para desconexão do usuário.

## 4.1 Telas padrões do sistema *NoCatAuth*

As telas padrões se resumem em quatro, sendo a primeira tela considerada a tela inicial do sistema, onde será solicitado do usuário, o nome e senha, conforme pode-se observar na figura 4.2. Assim que o sistema de autenticação *NoCatAuth* é inicializado são apresentados os certificados digitais criptografados, gerados para o servidor *web Apache*, caracterizando uma conexão segura através da porta 443, conforme pode-se observar na figura 4.3. A exibição das informações do certificado digital podem ser solicitadas pelo cliente em caso de dúvidas de autenticidade, através do botão examinar certificado.

Saudacoes! Bem vindo a rede da Prefeitura de Sacramento.

Login:

Password:

Don't have an account? [Register here!](#)

 **nocat**  
AUTHENTICATION

**Figura 4.2:** Tela de autenticação de usuários no sistema *NoCatAuth*



Figura 4.3: Conexão Segura, com confirmação do certificado digital

Após a autenticação, é exibida a tela de boas-vindas e uma tela de *Splash*, contendo informações sobre o tempo de permanência do usuário no sistema, além da opção de desconexão, conforme pode-se observar na figura 4.4 .

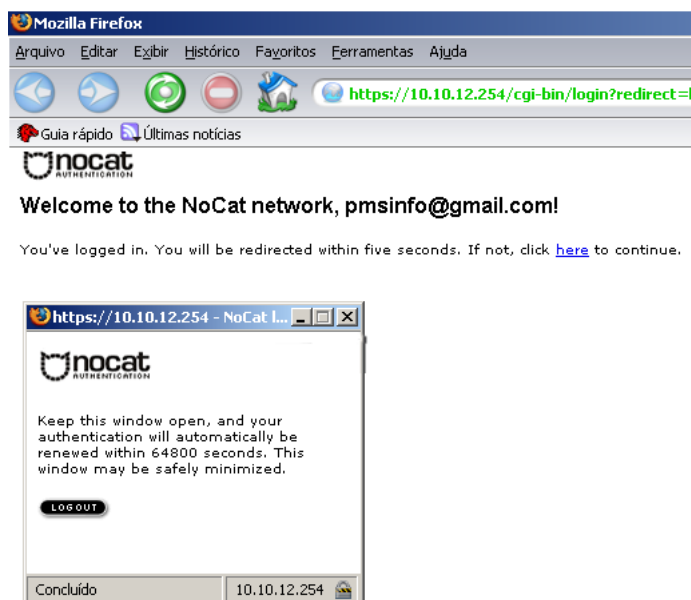
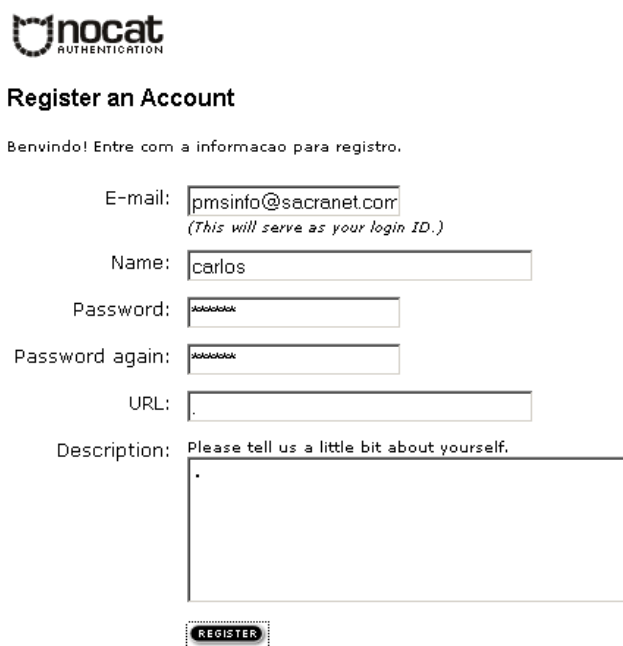


Figura 4.4: Tela de *Splash* contendo tempo de expiração e tela de boas-vindas



O cadastro de um novo usuário no sistema pode ser realizado através da utilização de um *link*, que é apresentado na tela de autenticação de usuários do sistema *NoCatAuth*, que redireciona o sistema para a tela de cadastro de novos usuários, conforme pode-se observar na figura 4.5. Para utilização do sistema *NoCatAuth*, em fins comerciais, este recurso deve apresentar-se oculta, ou protegida por senha, evitando assim que o próprio usuário possa realizar seu cadastro e ter acesso ao sistema.



The image shows a web form for registering an account. At the top left is the logo for 'nocat AUTHENTICATION'. Below the logo is the title 'Register an Account'. A message reads 'Bemvindo! Entre com a informacao para registro.' The form contains several input fields: 'E-mail:' with the value 'pmsinfo@sacranet.com' and a note '(This will serve as your login ID.)'; 'Name:' with the value 'carlos'; 'Password:' and 'Password again:' both with masked characters; 'URL:' with a single period '.'; and 'Description:' with the prompt 'Please tell us a little bit about yourself.' and a large empty text area. At the bottom of the form is a 'REGISTER' button.

**Figura 4.5:** Registro de usuários

Para o funcionamento correto da autenticação, o sistema *NoCatAuth* ao realizar a inclusão de um novo usuário, realiza validações nos campos chaves essenciais, para evitar que dados inválidos sejam aceitos. O campo representado pelo *email* do usuário, deve conter entre seus caracteres o símbolo @, além do campo senha possuir um valor.

A senha armazenada no banco de dados *MySQL*, apresenta-se armazenada no formato criptografado, evitando assim que em caso de invasão, alguma senha válida do sistema de autenticação possa ser usada por um usuário não cadastrado. O sistema *NoCatAuth*, é responsável por realizar a leitura padrão dos valores de senha

fornecidas pelo usuário final, transformando a senha em uma senha criptografada, realizando assim a comparação com o banco de dados, que por fim liberará ou não o acesso do usuário à *Internet*.

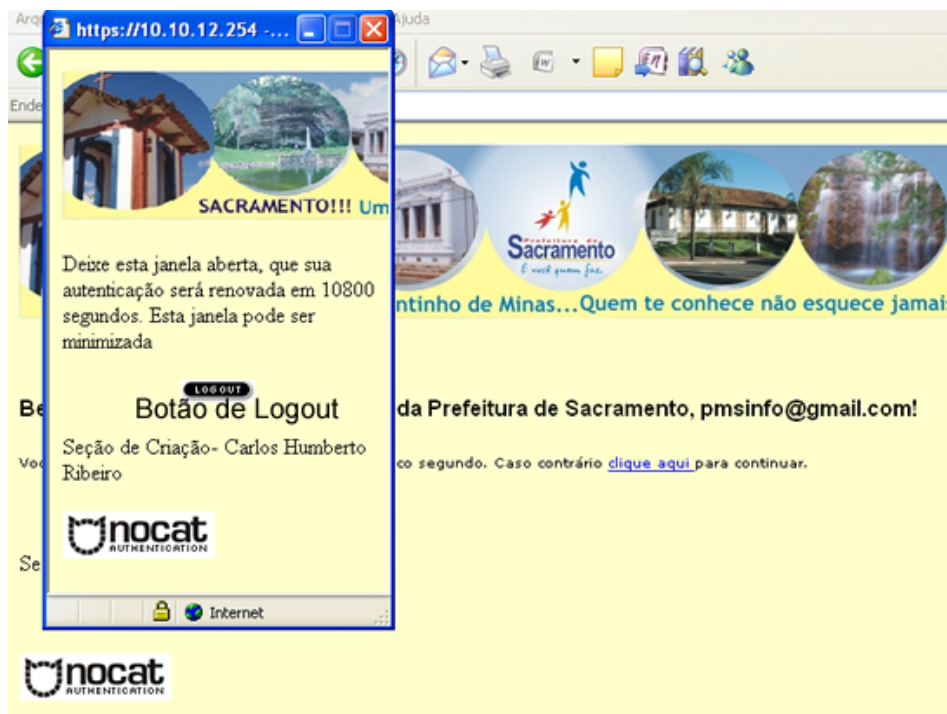
## 4.2 Alterações e implementações nas telas do sistema *No-CatAuth*

Para que o sistema *NoCatAuth* atinja a maturidade no campo comercial ou obtenha o potencial de um sistema de autenticação, torna-se necessário a realização de algumas modificações nas telas que o sistema apresenta para o usuário final, acrescentando informações consideradas essenciais para o usuário, além da modelagem da tela principal contendo a logomarca e novas funcionalidades. A tela inicial do sistema *NoCatAuth*, além de prever as modificações sugeridas acima, deve possibilitar que o usuário consiga realizar a saída do sistema através do botão *logout*, conforme pode-se observar na figura 4.6 .



**Figura 4.6:** Tela inicial do sistema *NoCatAuth*, ressaltando botão de *Logout*, informações ao usuário e padronização.

As modificações realizadas nas telas de registro e tela de *logout*, caracterizam-se apenas por modificações obrigatórias para personalização do sistema, onde torna-se necessário a adição de logomarca da empresa, cor característica da empresa, entre outras particularidades, conforme pode-se observar na figura 4.7 .

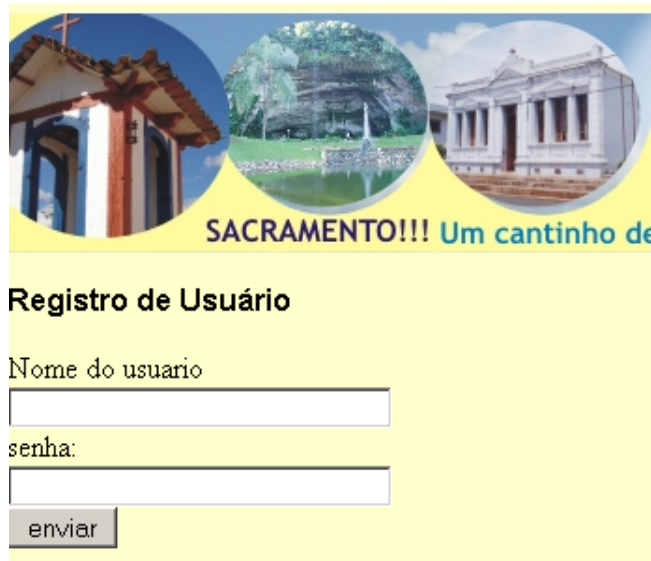


**Figura 4.7:** Tela de boas-vindas ao sistema de autenticação contendo a esquerda tela de *Splash*, com botão de *Logout*

O link utilizado para acesso a tela de registros, na tela de autenticação de usuários, na concepção inicial do sistema de autenticação *NoCatAuth*, deve ser removido, ou protegido através de senha de acesso, não permitindo assim a adição de novos usuários por um usuário final do sistema, o que torna o administrador o responsável pelo cadastro. O cadastro de novos usuários possui um sub-sistema, que exige o nome e senha do usuário administrador, que será responsável pela criação de novos usuários ao sistema, não permitindo o acesso a tela de registro de usuários sem um nível mínimo de segurança.

Quando o registro de um novo usuário do sistema *NoCatAuth* é solicitado, uma tela em formato *HTML*, é acionada recebendo o nome e senha do usuário administrador do sistema, garantindo assim que apenas o administrador realize o cadastro de novos usuários, conforme pode-se observar na figura 4.8. O cadastro de novos

usuários na sua concepção original, permite que página de registro seja acessada através da *URL*, não ocorrendo a proteção com senha e posterior redirecionamento para página de *autenticação*, caso o usuário não esteja cadastrado como usuário administrador do sistema.



SACRAMENTO!!! Um cantinho de

### Registro de Usuário

Nome do usuario

senha:

enviar

**Figura 4.8:** Tela de segurança do sistema de autenticação *NoCatAuth*.

### 4.3 Considerações Finais

As implementações propostas visam proporcionar ao sistema *NoCatAuth*, a maturidade necessária como um *software* de autenticação segura, e que mostrou-se como uma solução viável para autenticação em usuários em redes de topologia variada. No cenário atual, existem vários produtos similares que atingem o mesmo objetivo, inclusive provenientes do próprio sistema de autenticação *NoCatAuth*. Várias mudanças ainda podem ser implementadas, adicionando ao software um número maior de funcionalidades, como por exemplo a vinculação dos arquivos de log gerados pelo *Squid*.

A implantação do sistema de autenticação *NoCatAuth*, possibilitou que a instituição envolvida interligasse todos os pontos externos do prédio municipal a um servidor de *gateway*, evitando que vários pontos de acesso à *Internet* fossem solicitados, economizando 13 pontos de conexão. Houve também economia com a locação de vários pontos de distribuição de acesso à *Internet* podendo ser revertida

em equipamentos de informática para o uso da municipalidade, baseando que o aluguel é pago mensalmente, e o custo das antenas para interligação é um preço único. Os dois *links* de distribuição de acesso à *Internet*, antes da interligação, já possuíam a quantidade necessária de banda para abranger o sistema na sua totalidade, não necessitando assim de nenhuma modificação. Os *links* eram utilizados para acesso à *Internet* e *backup* do *link* principal. O sistema de autenticação baseado em *Linux* usando o sistema *NoCatAuth*, possibilitou a economia de recursos financeiros com licenças utilizados em programas proprietários, sendo revertido em benefícios e melhorias em equipamentos do setor de informática.

## Capítulo 5

# Conclusões e trabalhos futuros

A leitura deste trabalho possibilita a implementação de uma ferramenta relativamente simples, mas muito eficiente no propósito a que se destina: controlar o acesso à *Internet*, efetuado por usuários de redes de computadores utilizando rede sem fio ou rede cabeada, além de proporcionar um maior conhecimento sobre o sistema operacional *Linux*.

A implantação do sistema de autenticação *NoCatAuth*, possibilitou que a instituição envolvida interligasse todos os pontos externos do prédio municipal a um servidor de *gateway*, evitando que vários pontos de acesso à *Internet* fossem solicitados, economizando 13 pontos de conexão. A economia gerada através da redução do aluguel dos pontos de acesso à *Internet*, pode ser revertida em equipamentos de informática para o uso da municipalidade, baseando que o aluguel é pago mensalmente, e o custo das antenas para interligação é um preço único. Os dois *links* de distribuição de acesso à *Internet*, antes da interligação, já possuíam a quantidade necessária de banda para abranger o sistema na sua totalidade, não necessitando assim de nenhuma modificação. Os *links* eram utilizados para acesso à *Internet* e *backup* do *link* principal. O sistema de autenticação baseado em *Linux* usando o sistema *NoCatAuth*, possibilitou a economia de recursos financeiros com licenças utilizados em programas proprietários, sendo revertido em benefícios e melhorias em equipamentos do setor de informática.

A vantagem da utilização deste método de autenticação, paralelo a outros sistemas operacionais, é que torna possível controlar de maneira organizada e centralizada, mostrando ao usuário todas as informações necessárias ao seu funcionamento do sistema de autenticação, sem custos financeiros com programas proprietários para este fim, apenas utilizando de *Software Livre* e o sistema operacional *Linux*.

Desta forma, é impensável que no mundo atual um administrador de sistema desconsidere ou apenas desconheça o *Linux* como uma das melhores opções para implementação de servidores. Esta vantagem se dá justamente em função das características acima exemplificadas, além de outras vantagens como segurança, melhor desempenho, menor custo, melhor aproveitamento de *hardware* entre outras particulares.

O controle de acesso à *Internet* no mundo atual, mostra-se como uma necessidade a ser implementada em cada instituição ou corporação, pois através deste controle torna-se possível a economia do tempo das pessoas, considerado hoje escasso, bem como um melhor aproveitamento do tempo, além da diminuição de gastos com pagamento de melhorias no acesso a *Internet*. Diante da acirrada concorrência do mercado, as instituições devem buscar todas as formas para otimizar sua eficiência operacional, ou seja, produzir mais com menor custo. Certamente funcionários acessando a *Internet* de forma descontrolada em nada contribui para o alcance destes objetivos.

Desta forma, a versão do sistema de autenticação *NoCatAuth*, com as alterações propostas e implementadas, apresenta-se como uma boa opção para implementação de controle de acesso de redes cabeadas ou redes sem fio à *Internet*, atingindo todos os objetivos específicos propostos, liberando o acesso ao recurso só a usuários autenticados, garantindo assim a segurança de banda em redes sem fio, ou simplesmente o controle dos usuários que devem utilizar deste recurso dentro de uma instituição.

Como recomendações para trabalhos futuros, seria interessante a vinculação dos arquivos de *logs*, gerados pelo *Squid* com informações extraídas da autenticação de usuários, criando um módulo para armazenamento do horário de *login* e horário de *logoff* de cada usuário. Outra melhoria no sistema, considerado como trabalho futuro, é a melhoria no acesso a tela de cadastro de usuários, adicionando uma segurança adicional, e liberação do usuário administrador através de senha gravada em banco de dados *MySQL*.

# Referências Bibliográficas

ANDREASSON, O. *Iptables Tutorial 1.2.0*. [S.l.], 2005. Disponível em: <<http://iptables-tutorial.frozentux.net/iptables-tutorial.html>>. Acesso em: 10 Fev. 2007.

BRANDAO, P. *NoCatAuth - Construindo um firewall/gateway autenticado*. [S.l.], 2004. Disponível em: <<http://www.vivaolinux.com.br/artigos/verArtigo.php?codigo=1338pagina=2>>. Acesso em: 15 Maio 2007.

BRANDÃO, P. *MyAuth Gateway 2 - Manual*. [S.l.], 2004. Disponível em: <[http://201.78.74.2:808/?console=paginaerline\\_myauthgateway2\\_download](http://201.78.74.2:808/?console=paginaerline_myauthgateway2_download)>. Acesso em: 15 Abr. 2007.

DUFF, B. B. H. *Dominando Linux Red Hat e Fedora*. [S.l.]: Pearson Education do Brasil, 2003.

FILHO, J. E. M. *Firewall com Iptables*. [S.l.], 2004. Disponível em: <[www.eriberto.pro.br/iptables](http://www.eriberto.pro.br/iptables)>. Acesso em: 15 Fev. 2007.

MARCELO, A. *Squid Configurando o Proxy para Linux*. [S.l.], 2005.

MILANI, A. *Mysql Guia do Programador*. [S.l.]: Novatec, 2006.

NEMETH, E.; SNYDER, G.; HEIN, T. R. *Manual Completo do Linux*. São Paulo: Pearson Makron Books, 2004.

NETCRAFT. *Utilização de Servidores Web em Março de 2007*. [S.l.], 2007. Disponível em: <[http://news.netcraft.com/archives/2007/04/02-april\\_2007\\_web\\_server\\_survey.html](http://news.netcraft.com/archives/2007/04/02-april_2007_web_server_survey.html)>. Acesso em: 26 Abr. 2007.

NETO, U. *Linux Firewall Iptables*. [S.l.: s.n.], 2004.

SIELVER, S. S. . N. P. E. *Perl Guia Completo - Manual de Referência rápida*. [S.l.]: O'REILLY, 1999.



STANGER, J.; LANE, T. P. *Rede Segura Linux*. Rio de Janeiro: Alta Books, 2002.

STREBE, M.; PERKINS, C. *Firewalls*. São Paulo: Makron Books, 2002.

SUP la freeduc. *Tutoriel sur les serveurs*. [S.l.], 2004. Disponível em: <<http://www.linux-france.org/prj/edu/archinet/systeme/>>. Acesso em: 20 Maio 2007.

UCHÔA, J. Q. *Segurança Computacional*. 2. ed. Lavras: UFLA/FAEPE, 2005. (Curso de Pós-Graduação “Lato Sensu” (Especialização) a Distância: Administração de Redes Linux).

VISOLVE. *Squid Configuration Manual*. [S.l.], 2006. Disponível em: <[http://squid.visolve.com/squid/squid24s1/access\\_controls.htm](http://squid.visolve.com/squid/squid24s1/access_controls.htm)>. Acesso em: 02 Fev. 2007.

## Apêndice A

# Instalação do servidor *web Apache*

### A.1 Verificando a existência do servidor *web Apache*

A instalação do servidor *web Apache*, envolve vários passos que devem ser executados em servidor baseado no sistema *Linux*, onde a primeira fase trata-se da verificação se o serviço não se encontra instalado no equipamento em questão. Esta verificação é realizada pelo comando *Linux* “*service*” para sistemas baseados na distribuição *Fedora Core*.

O comando “*service*” é responsável por verificar se o serviço consultado no servidor *Linux* encontra-se disponível e ativo no sistema, bem como o *status* do mesmo, caso esteja instalado. O servidor *web Apache*, no sistema *linux*, é conhecido pela sigla *httpd*, onde invocado com comando *Linux* “*service httpd status*”, retornará o seu estado de funcionamento, podendo retornar entre os valores ativo, não ativo e não instalado.

### A.2 Instalando o servidor *web Apache*

Para instalar o servidor *web Apache*, é necessário que o usuário logado tenha poderes de administrador do sistema, ou tenha obtido, em caso de usuário comum do sistema, pelo comando *Linux* “*su -*”, que além dos poderes de administrador habilita as variáveis do usuário *root* para utilização do usuário comum. Assim que o comando “*su*” é acionado, será solicitada a senha do usuário administrador, que após aceita, transforma o usuário comum em administrador do sistema. Outra opção mais simples, trata-se de estar autenticado como usuário administrador

do sistema, conhecido como *root* em todas as distribuições baseadas no sistema *Linux*.

A instalação do servidor *web Apache*, na versão recente, bem como a atualização, em distribuições baseadas em *Fedora Core*, pode ser realizada através do utilitário do *Linux yum* recebendo como complemento os argumentos *install* para instalação ou *update* para atualização. Em caso de instalação do sistema, o comando “*yum*”, deve receber como argumento a palavra chave *install*, seguida do pacote, ao qual deseja-se instalar no sistema, *httpd*. Para que esta instalação seja possível torna-se necessário o servidor estar conectado a *Internet*, para receber e instalar o pacote desejado. O servidor *web Apache*, também pode ser instalado através dos pacotes distribuídos nas mídias de instalação, através do comando *Linux “rpm -Uvh pacote”*, onde o parâmetro *-U*, realiza a instalação ou atualização do pacote fornecido como complemento do comando.

### **A.3 Iniciando o servidor *web Apache***

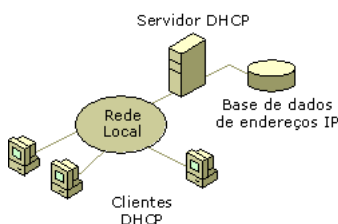
O servidor *web Apache*, pode ser iniciado em sistemas baseados na distribuição *Fedora Core*, através do comando “*service httpd start*”. Como opção este serviço pode ser configurado para ser acionado na inicialização do sistema *Linux*, através da comando “*ntsysv*”, com a posterior seleção do serviço *httpd* no *menu* exibido. Este comando apenas habilita a inicialização automática do serviço, assim que o equipamento carrega o sistema operacional, sendo necessário assim, após a configuração, o reinício equipamento.

## Apêndice B

# Configuração do Servidor de *Dhcp*

O protocolo de configuração dinâmica de cliente (*Dhcp*), um padrão do conjunto de protocolos *TCP/IP*, destinado a simplificar a gestão da configuração *IP* do cliente. A norma *Dhcp* permite a utilização de servidores *Dhcp* como forma de gerir a atribuição dinâmica de endereços *IP* e outros detalhes relacionados com a configuração para clientes ativados por *Dhcp* na rede.

Cada computador numa rede *TCP/IP* necessita de um endereço *IP* exclusivo. O endereço *IP* identifica o computador cliente e a sub-rede aos quais está ligado. Quando o computador é retirado para uma sub-rede diferente, o endereço *IP* tem de ser alterado, conforme pode-se observar na figura B.1.



**Figura B.1:** Estrutura de uma rede com servidor *Dhcp* (SUP, 2004)

A configuração do serviço de *Dhcp* em um servidor *Linux*, é baseado na edição e inclusão de parâmetros no arquivo `dhcpd.conf`, localizado no diretório `/etc`, em qualquer distribuição baseada em *Linux*, conforme pode-se observar na figura B.2.

```

ddns-update-style none;
default-lease-time 600;
max-lease-time 7200;
authoritative;
subnet 10.10.10.0 netmask 255.255.255.0{
    range 10.10.10.100 10.10.10.200;
    option routers 10.10.10.1;
    option domain-name-servers 200.243.249.132,200.243.249.134
}

```

**Figura B.2:** Configuração de um servidor *Dhcp* através do arquivo */etc/dhcpd.conf*

O arquivo de configuração do servidor de *Dhcp*, referenciado como *dhcpd.conf*, necessita dos seguintes parâmetros para prover o serviço de *IP* dinâmico dentro de uma rede da mesma classe:

- `ddns-update-style none` . Esta opção serve para indicar se o servidor *DNS* será atualizado quando um aluguel de *IP* for solicitado.
- `default-lease-time` . Esta opção define o período em segundos, para aluguel de um endereço de *IP*.
- `max-lease-time`. Esta opção define o período máximo de aluguel, em segundo de um endereço de *IP*.
- `subnet netmask` . Esta opção define a rede de atuação e máscara, para os alugueis de endereço de *IP*.
- `range`. Esta opção define o escopo dos endereços disponíveis para transmissão dinâmica.
- `option routers` . Esta opção define o *router* padrão, popularmente conhecido como *Gateway padrão*.
- `option domain-name-servers` . Esta opção define os endereços dos servidores de *DNS*.
- `broadcast-address` . Esta opção define a transmissão de endereços.

## Apêndice C

# Instalação do módulo de acesso a banco de dados *MySQL* em modo gráfico *PhpMyAdmin*

O módulo de administração remota a bancos de dados, através da utilização de um *Browser* de *Internet*, *PhpMyAdmin*, pode ser obtido, por meio de *download*, através de um dos *mirrors* do *SourceForge*,<sup>1</sup> repositório de dados de projetos *OpenSource*. Após a obtenção do arquivo de instalação para distribuição *Linux*, o administrador deve descompactá-lo em uma pasta de sua preferência, por se tratar de um arquivo com o padrão `tar.gz`. Estes arquivos são gerados pelo meio de compactação através do comando `tar` usando o compactador `gzip`. O processo de descompactação e modificação dos parâmetros necessários ao funcionamento do módulo *PhpMyAdmin* pode ser observado na figura C.1.

```
tar -xvzf phpMyAdmin-xxx.tar.gz
mv phpMyAdmin-xxx phpmy
cd phpmy/
mv config.inc.sample.php config.inc.php
chmod 644 phpmy/config.inc.php
```

**Figura C.1:** Instalação do módulo de administração de banco de dados *PhpMyAdmin*

O módulo *PhpMyAdmin*, pode ser obtido em várias versões diferentes, sendo a versão atual a de número 2.10.2. A diferença encontrada entre as diversas versões do módulo *PhpMyAdmin*, está em algumas características adicionais e melhorias,

---

<sup>1</sup><http://sourceforge.net/projects/phpmyadmin/>

sendo assim possível a administração do banco de dados, em uma versão anterior a acima exposta.

Após a descompactação dos arquivos do módulo *PhpMyAdmin*, o diretório deve ser movido para a pasta raiz, onde são exibidas e executadas as páginas de *Internet* pelo servidor *web Apache*, localizada no sistema *Linux* através da pasta `/var/www/html`. O nome padrão da pasta onde se encontra-se localizado o arquivos do módulo de administração *PhpMyAdmin* deve ser modificada, pois a mesma será acessada pelo navegador de *Internet*, pelo nome da pasta localizada dentro do diretório do servidor *web Apache*.

A configuração do módulo *PhpMyAdmin* é realizada através de um único arquivo denominado `config.inc.php`, que deve ter suas propriedades de leitura e execução modificadas, evitando assim acessos indevidos. Esta modificação é realizada pelo comando *Linux* “*chmod*”, usando os parâmetros `644`, onde será permitida a leitura e gravação para o dono do arquivo e apenas a leitura para os demais usuários e grupo do dono do arquivo. Nas versões mais recentes do produto, o arquivo `config.inc.php` possui o nome `config.inc.sample.php`, considerado como um arquivo de exemplo das configurações do módulo, sendo necessário assim a mudança de nome para `config.inc.php`.

O funcionamento do módulo de administração de Banco de dados *PhpMyAdmin*, através de um *web Browser*, pode ser observado na figura C.2.



Figura C.2: Tela com funcionalidades do sistema *PhpMyAdmin*

## Apêndice D

# Instalação e configuração do servidor de banco de dados *MySQL*

O servidor de banco de dados *MySQL* conhecido no sistema *Linux*, em particular na distribuição *Fedora Core* através do serviço *MySQLD*, pode ser instalado, como os demais pacotes do sistema, das seguintes maneiras:

- Através do utilitário de instalação de programas *yum*.
- Pela seleção prévia na instalação inicial da distribuição, selecionado, entre os pacotes a instalar , o serviço *MySQLD*.
- Pela mídia que acompanha a distribuição através do comando *rpm*, em particular na distribuição *Fedora*, seguida pelo nome do pacote a instalar.

A instalação de qualquer serviço em distribuições baseadas no sistema operacional *Linux*, depende das permissões que o usuário possui. Um usuário comum, necessita receber os poderes de usuário administrador, para realizar a instalação de qualquer aplicativo dentro do sistema *Linux*. O usuário comum, obtém os poderes de administrador do sistema através do comando *Linux* “*su-*”, sendo solicitado a senha do superusuário, para realizar a mudança das permissões.



## D.1 Verificando a existência do servidor de banco de dados *MySQL*

A instalação do servidor de banco de dados *MySQL*, envolve vários passos que devem ser executados em servidor baseado no sistema *Linux*, onde a primeira fase trata-se da verificação se o serviço não se encontra-se instalado no equipamento em questão. Esta verificação é realizada pelo comando *Linux* “*service*” para sistemas baseados na distribuição *Fedora Core*.

O comando “*service*” é responsável por verificar se o serviço consultado no servidor *Linux* encontra-se disponível e ativo no sistema, bem como o *status* do mesmo, caso esteja instalado. O servidor de banco de dados *MySQL*, no sistema *linux*, é conhecido pelo serviço *MySQLD*, onde invocado com comando “*service mysqld status*”, retornará o seu estado de funcionamento, podendo retornar entre os valores ativo, não ativo e não instalado.

## D.2 Instalando o servidor de Banco de dados *MySQL*

Como mencionado anteriormente, somente ao usuário administrador é permitido a instalação e mudanças de configurações do sistema, e somente ele tem direitos e poderes para a instalação. A instalação do serviço *MySQL* envolve a instalação de dois pacotes, o pacote *MySQL Client* responsável pela interação através de seus programas com o serviço de banco de dados, e o pacote *MySQL Server*, que trata-se do banco de dados propriamente dito, conforme pode-se observar na figura D.1.

```
yum install MySQL
yum install MySQL-server
```

**Figura D.1:** Instalação do servidor de banco de dados *MySQL* através do utilitário *yum*

## D.3 Iniciando o servidor de banco de dados *MySQL* e configurações Adicionais

O servidor de banco de dados *MySQL*, pode ser iniciado em sistemas baseados na distribuição *Fedora Core*, através do comando “*service mysqld start*”. Como opção este serviço pode ser configurado para ser acionado na inicialização do sistema *Linux*, através da comando “*ntsysv*”, com a posterior seleção do serviço *MySQLD* no *menu* exibido. Este comando apenas habilita a inicialização automática do ser-

viço, assim que o equipamento carrega o sistema operacional, sendo necessário assim, após a configuração, o reinício equipamento.

O simples fato do serviço de banco de dados estar em funcionamento, não garante que o serviço esteja apto a receber informações no formato de dados *MySQL*. Após a ativação do serviço de banco de dados torna-se necessário a instalação da base inicial do serviço *MySQL*, através do comando *Linux* ```mysql_install_db```.

Por padrão em distribuições *Linux* baseadas em fedora core, o diretório padrão que armazena as tabelas e banco de dados do servidor *MySQL*, criados após a execução do comando *Linux* ```mysql_install_db``` é `/var/lib/mysql`. Por motivos de segurança este diretório e arquivos contidos no seu interior, devem pertencer ao usuário e grupo *mysql*, possuindo permissão de leitura e escrita apenas ao dono do arquivo e grupo a que o dono do arquivo pertence, não permitindo assim a leitura a usuário normais. Estas configurações já apresentam pré-configuradas após a instalação do serviço e criação das tabelas padrões. Os comandos *Linux* responsáveis pelas mudanças são: `“chmod 660 /var/lib/mysql -R”` e `“chown mysql:mysql /var/lib/mysql -R”`.

A instalação da base de dados inicial do sistema de banco de dados *MySQL*, não define inicialmente uma senha para superusuário que terá os poderes de administração e modificação dentro do sistema. Após a instalação do sistema base, uma senha de superusuário deve ser definida para o banco, evitando assim falhas de segurança, como invasão do banco de dados e roubo de informações sigilosas. A definição da senha do banco de dados para o superusuário pode ser obtida pelo comando *Linux* ```mysqladmin password senhadoperusuario```.