

## Revisão Completa dos Tópicos do Exame 202-450

### Tópico 207 – DNS (Domain Name Server)

#### 207.1 – Configuração Básica de um Servidor DNS

##### Conceitos e Termos Importantes:

- NS (Name Server): Armazena informações sobre uma parte do Domain Name Space, também chamado de zona.
- Root Domain: Referente ao domínio/zona raiz do DNS, representando pelo . (ponto). Os NSs responsáveis pelo Root Domain são chamados de Root Servers.
- TLD – Top Level Domains: Domínios imediatamente abaixo da raiz (.), por exemplo .com, .br, .net, .org e etc.
- DNS Resolver
  - Software ou biblioteca responsável por fazer a consulta de DNS
  - Utilizado no sistema local (DNS Client) e também é parte do DNS Server
  - Pode armazenar os resultados em cache
- BIND (Berkeley Internet Domain Server)
  - Open Source DNS Server
  - Implementa o protocolo DNS
  - Implementação de DNS Server mais utilizada
- Alternativas ao BIND
  - djbdns - Implementação DNS criada por Daniel J. Bernstein
  - dnsmasq – Combinação leve de um DNS Caching com DHCP
  - PowerDNS – Implementação DNS de grande porte. “Concorrente” do BIND

---

##### Tipos de NS:

- Primary (Master) : Servidor que possui autoridade sobre o domínio, definindo todas as informações sobre esta zona de DNS. No registro de um domínio, sempre deve haver ao menos um dos NS como master.
- Secondary (Slave) : O NS Slave transfere para si as informações definidas para um domínio a partir de um NS Master. Dessa forma as informações também ficam armazenadas no servidor e assim também possui autoridade sobre o domínio.
- Caching : O Servidor do tipo caching é capaz de fazer pesquisas recursivas para fazer a resolução DNS e armazena esse resultado em cache.
- Forwarding: Nesse caso o NS encaminha (delega) as pesquisas para outro servidor. Após receber a resposta o resultado também é armazenado em cache.

---

## **Arquivos e Diretórios de Configuração**

Distribuições baseadas em RedHat seguem o padrão do BIND:

- **/etc/named.conf** : Arquivo de configuração principal
- **/var/named/** : Diretório que armazena os arquivos de zona e outros registros

Distribuições baseadas em Debian armazenam os arquivos de configuração no diretório **/etc/bind**.

## **Principais Configurações (seção options):**

- **listen-on** : define a porta e os IPs que vão receber conexões no servidor DNS
  - O DNS utiliza a porta 53 e em geral o protocolo UDP, o protocolo TCP é usado apenas para transferências de zona.
- **directory**: indica o diretório base do servidor, normalmente o **/var/named/**
- **recursion**: indica se o servidor faz ou não pesquisas recursivas (resolução de nomes de outros domínios)

---

## **Configurações de um Servidor do tipo Caching-Only**

```
zone "." IN {  
    type hint;  
    file "named.ca";  
};
```

---

## **Configurações de Logging**

Nas configurações de logs do servidor DNS, duas definições são importantes:

- **channel** – Define o local em que o log será registrado e qual nível/severidade será adotado

- **category** – Define o que será logado, indicando-se também o canal (channel) que será utilizado.

Lista de tipos de severidade e categorias: <http://www.zytrax.com/books/dns/ch7/logging.html>

Exemplo:

```
logging {
    channel ricardo {
        file "data/ricardo.log";
        severity dynamic;
    };
    category queries {
        ricardo;
    };
};
```

---

## **Principais Comandos**

- **rndc** – Utilizado para realizar operações no servidor DNS. Principais opções:
  - reload : Recarrega todas as configurações, tanto do named.conf quanto de todas as zonas configuradas
    - reload <domínio> : Recarrega apenas as configurações de um domínio específico
  - reconfig : Recarrega as configurações do named.conf e de novas zonas de DNS criadas
  - flush : Limpa o cache do servidor DNS
  - retransfer <domínio> : Força a transferência de zona de um domínio em um servidor atuando como slave.
- **host** – Faz consultas DNS. Formas de uso:
  - # host www.lpi.org : Utiliza o servidor DNS configurado no ambiente
  - # host www.lpi.org 192.168.1.220 : Utiliza o servidor de DNS 192.168.1.220
  - # host -t MX lpi.org : Obtém o registro do tipo MX do dominiolpi.org
- **dig** – Faz consultas DNS mais elaboradas. Formas de uso:
  - # dig www.lpi.org
  - # dig www.lpi.org @192.168.1.220
  - # dig -t MX lpi.org
- **kill** – Com o sinal 1 (SIGHUP) pode ser utilizado para que o processo do BIND releia todas suas configurações:
  - # kill -1 PID

- **named-checkconf** – Valida a sintaxe das configurações do /etc/named.conf

## 207.2 – Criar e Manter Zonas de DNS

### Domínio do Tipo Master:

No /etc/named.conf:

```
zone "dominioexemplo.com.br" IN {  
    type master;  
    file "dominioexemplo.zone";  
};
```

No arquivo de zona (/var/named/dominioexemplo.zone):

```
$TTL 3h  
@      IN      SOA  servidor.dominioexemplo.com.br. admin.dominioexemplo.com.br. (  
    2018032801 ; serial. Número utilizado para indicar mudanças na zona  
    28800      ; refresh. Após quanto tempo o NS slave deve verificar novamente por  
atualizações  
    7200       ; retry. Em caso de falha no refresh, após quanto tempo deve haver uma  
retentativa  
    2419200    ; expire. Validade das informações. Após quanto tempo as informações não  
atualizadas do slave deixarão de ser válidas  
    150        ; negative caching. Por quanto tempo uma resposta negativa fica em cache  
)  
       NS      servidor      ; name server  
       MX      5      mailserver ; mail exchange  
servidor A      192.168.1.5      ; glue record  
mailserver A    192.168.1.10  
www       CNAME servidor  
mail      CNAME mailserver
```

- \$TTL – Tempo de vida dos dados no cache de quem obter as informações
- @ indica o nome do domínio indicado no named.conf
- No registro SOA (Start of Authority), o primeiro endereço refere-se ao NS e o segundo ao e-mail do administrador
- O . sempre deve ser utilizado no final da referência ao FQDN (Fully Qualified Domain Name, ou endereço completo). Na falta do ., o BIND inclui o domínio automaticamente
- Os números indicam tempos em segundos, mas também podem ser utilizados **h** (horas), **d** (dias) ou **w** (semanas).
- Glue Record é um registro do tipo A que relaciona o nome do NS do domínio ao seu endereço IP

### Principais Tipos de Registros:

- A : Endereço IPv4
- AAAA : Endereço IPv6
- CNAME : Canonical Name (apelido)

- TXT : Texto
  - SOA : Start of Authority
  - NS : Name Server
  - MX : Mail Exchange (Servidor de E-mail). É acompanhado de um número em que quanto menor o valor, maior a prioridade entre os servidores de e-mail.
  - PTR : DNS Reverso
- 

### **Domínio do Tipo Slave**

No /etc/named.conf:

```
zone "dominioexemplo.com.br" {  
    type slave;  
    file "dominioexemplo.com.br.zone";  
    masters { 192.168.1.220; };  
};
```

- O slave transfere uma nova versão das informações a cada tempo de “refresh” (definido no registro SOA) e sempre que o serial no master for maior que no slave
- 

### **Forwarding**

No /etc/named.conf:

Para redirecionar para outro servidor DNS todas as consultas recursivas, deve ser incluída a seguinte configuração na seção “options”:

- forwarders { IP1; IP2; };

Para que o servidor encaminhe tanto as pesquisas recursivas quanto as pesquisas internas a outro servidor, deve ser incluída a configuração:

- forward only;

Para que apenas as consultas referentes a um domínio específico sejam encaminhadas, a zona deve ser criada com o tipo “forward”:

```
zone “dominioexemplo.com.br” IN {  
    type forward;  
    forwarders { IP1; IP2; };  
};
```

---

### **DNS Reverso**

Possibilita a descoberta de um nome de DNS a partir de um endereço IP.

Configuração no /etc/named.conf:

```
zone "1.168.192.in-addr.arpa" IN {  
    type master;  
    file "1.168.192.in-addr.arpa.zone";  
};
```

Arquivo de zona (/var/named/1.168.192.in-addr.arpa.zone)

[root@linux-centos named]# cat 1.168.192.in-addr.arpa.zone

\$TTL 3h

```
@    IN      SOA    servidor.dominioexemplo.com.br. admin.dominioexemplo.com.br. (  
    2018032801  ; serial  
    28800      ; refresh  
    7200       ; retry  
    2419200    ; expire  
    150        ; minimum  
    )  
    NS        servidor.dominioexemplo.com.br.  
5     PTR     servidor.dominioexemplo.com.br.  
10    IN      PTR   mailserver.dominioexemplo.com.br.
```

# host 192.168.1.5

5.1.168.192.in-addr.arpa domain name pointer servidor.prudenciato.com.br.

---

## **Comandos Relacionados**

- **named-checkzone** <dominio> <arquivo-zona> : Verifica a sintaxe do arquivo de zona
  - # named-checkzone dominioexemplo.com.br /var/named/dominioexemplo.zone
- **named-compilezone** : Converte um arquivo de zona slave em formato texto legível
  - # named-compilezone -f raw -F text -o saida.txt exemplo.com.br exemplo.zone
- **dig** : Pode ser utilizado para buscar informações referentes à transferência de zonas com o tipo axfr:
  - # dig @192.168.1.220 axfr dominioexemplo.com.br

## 207.3 – Segurança no Servidor DNS

### Considerações Importantes para a Segurança do Serviço

- O serviço BIND deve estar sempre atualizado
- O processo do BIND (named) não pode ser executado pelo usuário root
- Em ambientes críticos o servidor DNS não deve compartilhar o servidor com outros serviços
- Considerar a separação (split) do serviço de DNS de acordo com o cenário. Ter servidores (ou views) diferentes para tipos de requests diferentes, por exemplo Internet e Intranet.

---

### Configurações do named.conf para Segurança

- version “hidden” : Esconder a versão do software
- backhole : IPs/Redes que não serão respondidos pelo servidor
- allow-query : IPs/Redes que podem fazer consultas no servidor
- allow-recursion : IPs/Redes que podem fazer consultas recursivas no servidor
- allow-transfer : IPs/Redes que podem realizar operações de transferência de zonas
- acl : definir grupos de IPs/Redes
- view : Definir grupos de regras e declarações

---

### TSIG – Transaction Signature

Utiliza uma **chave simétrica compartilhada** para aumentar a segurança na **comunicação entre servidores** DNS. Essa chave é utilizada para se **autorizar o acesso** às informações de um servidor. Muito utilizado para proteger a comunicação entre servidores master e slave.

O comando **dnssec-keygen** é utilizado para gerar as chaves. Exemplo:  
# dnssec-keygen -a HMAC-MD5 -b 256 -r /dev/urandom -n HOST chaves

#### Exemplo de Configuração no Master:

```
key exemplo {  
    algorithm HMAC-MD5;  
    secret “xxxxxxxxxxxxxxxx”;  
};
```

```
allow-transfer { key exemplo; };
```

#### Exemplo da Configuração no Slave:

```
key exemplo {  
    algorithm HMAC-MD5;  
    secret “xxxxxxxxxxxxxxxx”;  
};
```



```
server IP {  
    keys { exemplo; };  
};
```

---

### **DNSSEC (BIND DNS Security Extensions)**

Utiliza chaves assimétricas (públicas e privadas) para assegurar a autenticidade e integridade das respostas enviadas pelos servidores DNS.

Através da chave privada uma zona de DNS é assinada e a chave pública possibilita garantir a autenticidade da resposta.

As chaves são geradas pelo comando **dnssec-keygen**. Por exemplo:  
# dnssec-keygen -a DSA -b 1024 -r /dev/urandom -n ZONE exemplo

O arquivo Kexemplo.+999.+999999.key conterá a chave pública, que deve ser inserida como um registro na zona de DNS.

O arquivo da zona de DNS deve ser assinado com o comando **dnssec-signzone**, utilizando o arquivo de chave privada Kexemplo.+999.+999999.private. Exemplo:  
# dnssec-signzone -P -r /dev/urandom -o exemplo.com.br exemplo.zone Kexemplo.+999.+999999.private.

O comando dnssec-signzone gerará um arquivo de zona assinado (exemplo.zone.signed), que deve ser configurado no /etc/named.conf.

---

### **DANE (DNS -Based Authentication of Named Entities)**

Solução criada para resolver o problema dos CAs (Certification Authorities), criando uma forma de associar um domínio a um CA específico através de um registro do tipo TLSA inserido dentro do arquivo de zona.

---

### **Enjaulamento de DNS (chroot jail)**

Forma de executar o serviço de DNS em um ambiente isolado do resto do servidor, impedindo que brechas no serviço impactem a segurança do servidor como um todo.

Na prática, é criado um novo / exclusivo para o BIND, por isso chroot. Nessa nova raiz são instalados, criados e copiados todos os arquivos e diretórios utilizados pelo BIND, incluindo arquivos do /etc/, /lib, /sbin, /var/run e etc, em uma nova estrutura completa.

Na execução do processo bind, deve ser utilizada a opção -t para definir o diretório chroot, por exemplo:

```
# named -u bind -t /chroot/
```

# Tópico 208 – Web Services

## 208.1 – Implementando um Servidor Web (Apache)

### Arquivo de Configuração

Principal: /etc/httpd/conf/httpd.conf

Configurações extras: /etc/httpd/conf.d/

O pacote do apache em distribuições baseadas em Debian utilizam como principal arquivo de configuração o /etc/apache2/apache2.conf, mas a LPI e a documentação do Apache HTTPD mencionam o httpd.conf como sendo o arquivo de configuração principal.

As principais configurações do httpd.conf são:

- **ServerRoot** – Diretório base do apache, em geral o /etc/httpd/
- **Listen** – IP/Porta em que o Apache escutará as conexões
- **ServerAdmin** – Define um e-mail de administrador do servidor
- **ServerName** – Nome/Endereço e Porta que o Apache utiliza para se identificar
- **DocumentRoot** – Diretório em que se encontram os arquivos que serão servidos pelo servidor web

\* Outras configurações serão estudadas a seguir

---

### Processos e Comandos

#### Processo httpd

O processo padrão do Apache HTTPD é o **/usr/sbin/httpd**. Esse padrão é utilizado em distribuições baseadas em RedHat.

No Debian o processo é o **/usr/sbin/apache2**.

Tanto o “httpd” quanto o “apache2” disponibilizam uma série de opções, entre elas:

- -v : Exibir a versão do Apache
- -V : Exibir as configurações definidas na compilação do Apache
- -l : Exibir os módulos compilados com o Apache

#### Comando apache2ctl (apachectl)

Principal comando para administração do Apache HTTPD. Entre suas opções temos:

- status/stop/start/restart
- graceful – Reinicia o apache aguardando que as conexões ativas terminem
- graceful-stop – Pára o apache aguardando que as conexões ativas terminem
- configtest – Verifica se há erros de sintaxe nas configurações

## MPM (Multi-Processing Modules) – Gerenciamento de Processos e Performance

O MPM determina a forma como o Apache gerencia seus processos e as requisições que recebe.

As 3 principais opções são:

- **Prefork:** Nesse modo, o Apache possui um processo pai para gerenciamento e vários sub-processos para tratamento das requisições. Nesse caso o sub-processo trata uma requisição por vez.
- **Worker:** Assim como no prefork, nesse modo há um processo pai e vários sub-processos. A diferença é que cada sub-processo possui diversas threads e cada thread trata uma requisição por vez.
- **Event:** Muito similar ao worker, mas consegue gerenciar melhor as conexões dentro das threads.

Os principais parâmetros de configuração que podem ser introduzidos no httpd.conf para configurar o funcionamento do MPM são:

- **StartServers** – Número de instâncias do apache (processos filho) iniciados ao se iniciar o apache.
- **MaxClients** – Limite de conexões simultâneas que o Apache tratará.
- **MaxRequestsPerChild** – Número de requests que uma instância tratará antes de morrer.
- **MinSpareServers** (prefork) – Quantidade mínima de processos extras que serão mantidos pelo Apache
- **MaxSpareServers** (prefork) – Quantidade máxima de processos extras que serão mantidos pelo Apache
- **MinSpareThreads** (worker/event) – Quantidade mínima de threads extras que serão mantidos pelo Apache
- **MaxSpareThreads** (worker/event) – Quantidade máxima de threads extras que serão mantidos pelo Apache
- **ThreadsPerChild** (worker/event) – Quantidade de threads em cada processo filho.

---

## Módulos

O Apache é uma aplicação modular, ou seja, recursos podem ser incorporados através de módulos externos.

Esses módulos ficam normalmente localizados no diretório **/etc/httpd/modules** e possuem nomes que seguem o padrão **mod\_\*.so**.

Os módulos são carregados no Apache através do parâmetro LoadModule, como no exemplo:

- **LoadModule** php5\_module /etc/httpd/modules/libphp5.so

Dentre os módulos mais utilizados temos o mod\_php, para suporte à linguagem PHP, e o mod\_perl, para suporte à linguagem Perl. Nesses casos, as principais configurações são:

### **PHP:**

# Faz o carregamento do módulo

LoadModule php5\_module /etc/httpd/modules/libphp5.so

# Define o Handler (processo interno) que tratará arquivos que terminem com .php

<FilesMatch \.php\$>

**SetHandler** application/x-httpd-php

</FilesMatch>

# Inclui arquivos .php como tipo “text/html”. Também chamado de “Content Type” ou “Mime Type”

**AddType** text/html .php

# Inclui os arquivos de nome index.php na lista de possíveis Index considerados pelo Apache

**DirectoryIndex** index.php

## Perl:

# Faz o carregamento do módulo

LoadModule perl\_module modules/mod\_perl.so

Alias /perl /var/www/perl

<Directory /var/www/perl>

**SetHandler** perl-script

# Define o Handler

    PerlResponseHandler ModPerl::Registry

    PerlOptions +ParseHeaders

**Options +ExecCGI**

# Permite a execução através de CGI

</Directory>

---

## Configurações e Arquivos de Log

Os diretórios padrão de logs do Apache são o /var/log/httpd ou o /etc/httpd/logs, que normalmente é um link para o primeiro.

As seguintes configurações são adicionadas ao httpd.conf para definir como será o registro de logs:

- **ErrorLog** – Define o arquivo em que serão inseridos os registros de erro
  - ErrorLog "logs/error\_log"
- **LogLevel** – Define o nível das mensagens do ErrorLog
  - LogLevel debug
  - Os valores possíveis são: debug, info, notice, warn, error, crit, alert, emerg
- **LogFormat** – Define um formato de log de acessos. Define o que será registrado
  - LogFormat "%h %l %u %t \"%r\" %>s %b \"%{Referer}i\" \"%{User-Agent}i\"""  
combined
- **CustomLog** – Define o arquivo em que será registrado o formato definido no LogFormat
  - CustomLog "logs/access\_log" combined

---

## Controle de Acesso

Os recursos de controle de acesso também é realizado através de módulos. Os principais deles são:

- mod\_auth\_basic – Possibilita autenticação através de usuários e senhas
- mod\_authz\_core – Implementa uma base para autenticação e controle de acesso. Normalmente usado em conjunto com outros módulos
- mod\_authn\_file – Autenticação através de arquivos texto com senhas
- mod\_access\_compat – Controle de acesso através de IP e Hostname
- mod\_authz\_host – Controle de acesso através de IP e Hostname

## **Controle de Acesso por Usuário e Senha:**

### Geração do Arquivo de Senhas:

- # htpasswd -s -c .htpasswd usuario
  - -s : Uso do SHA1
  - -c: Cria um novo arquivo. Deve ser omitido para inserir novos usuários no mesmo arquivo
- Sem opções o htpasswd irá alterar a senha do usuário. Exemplo:
  - # htpasswd .htpasswd usuario

### Configurações do httpd.conf. Exemplo:

```
<Directory /var/www/html >
Options Indexes FollowSymLinks
AllowOverride all                # Permite o uso do arquivo .htpasswd
Require all granted
</Directory>

<Directory /var/www/html/topsecret>
AuthType Basic
AuthName "Area Secreta de Acesso"
AuthUserFile /var/www/html/topsecret/.htpasswd
AuthGroupFile /var/www/html/topsecret/.htgroup
# Require group suporte          # Exige que o usuário faça parte do grupo suporte
                                # definido no arquivo .htgroup
Require valid-user              # Exige que o usuário exista no arquivo .htpasswd
# Require user ricardo aluno1    # Libera acesso apenas para os usuários especificados
</Directory>
```

## **Controle de Acesso por IP/Rede/Host**

### Configurações do httpd.conf. Exemplo:

Modo “atual”, pelos módulos mod\_authz\_host e mod\_authz\_core.

```
<Directory /var/www/html/admin>
Require ip 192.168.1.210
Require ip 192.168.1.1
Require ip 172.16
Require ip 10.0.0.0/16
```

```
Require host dominioexemplo.com.br
</Directory>
```

Modo “antigo”, pelo módulo mod\_access\_compat

```
<Directory /var/www/html/admin>
  Order Deny,Allow
  Deny from all
  Allow from 192.168.1.210
  Allow from 192.168.1.1
</Directory>
```

---

## **Redirecionamentos**

- **Redirect**
  - O Apache diz ao cliente requisitar outra URL.
  - Redirecionamento do lado do cliente (client-side)
  - Exemplo: Redirect /google http://www.google.com.br
  - Exemplo: Redirect /antigo.html http://www.dominioexemplo.com.br/teste.php
- RedirectMatch – O mesmo que o Redirect mas possibilita o uso de expressões regulares
- **Alias**
  - Redirecionamento é feito internamente no servidor (server-side)
  - Pode levar a um diretório fora do DocumentRoot
  - Exemplo: Alias /docs /var/www/docs
- AliasMatch – O mesmo que o Alias mas possibilita o uso de expressões regulares

---

## **UserDir**

Possibilita que cada usuário do servidor possua uma área que será disponibilizada pelo Apache.

Configuração no httpd.conf:  
UserDir public\_html

Nesse caso, os arquivos devem estar localizados no diretório:  
/home/user/public\_html/

O acesso será feito pela URL:  
http://example.com/~user

---

## **VirtualHost**

Possibilita a hospedagem de diversos sites no mesmo servidor.

O site/host virtual pode ser identificado pelo IP e pelo Nome:

- **Baseado no IP:** Cada IP da máquina é associado a um conteúdo web, a um DocumentRoot.
- **Baseado no Domínio:** Nesse caso, cada URL é associada a um conteúdo. Para isso o servidor Web identifica o domínio através do request feito pelo cliente (navegador). Modo mais comum de uso.

### Exemplo de Configuração Baseada em IP:

Listen 192.168.1.100:80

Listen 192.168.1.200:80

<VirtualHost 192.198.1.100>

**DocumentRoot** /var/www/html/teste1

</VirtualHost>

<VirtualHost 192.198.1.200>

**DocumentRoot** /var/www/html/teste2

</VirtualHost>

### Exemplo de Configuração Baseada em Nome:

Listen 192.168.1.100:80

<VirtualHost 192.198.1.100>

**ServerName** [www.teste1.com.br](http://www.teste1.com.br)

**DocumentRoot** /var/www/html/teste1

</VirtualHost>

<VirtualHost 192.198.1.100>

**ServerName** [www.teste2.com.br](http://www.teste2.com.br)

**DocumentRoot** /var/www/html/teste2

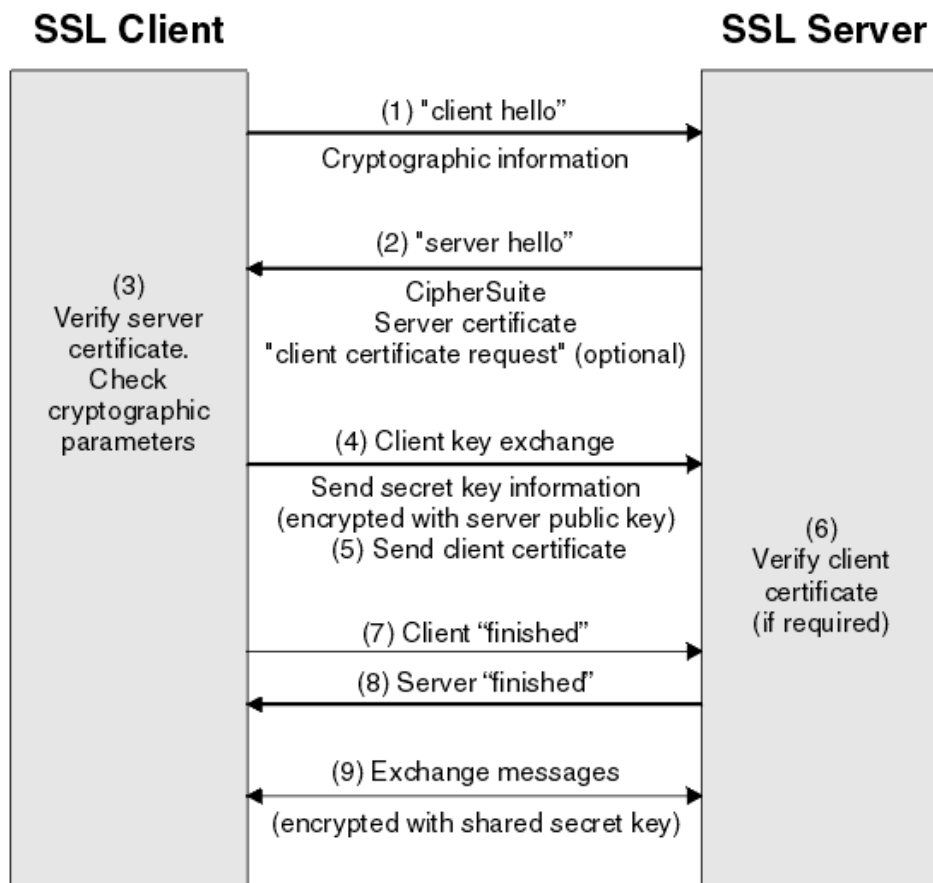
</VirtualHost>

## 208.2 – Configurações do Apache para HTTPS

Implementação de protocolos de criptografia para encriptação e autenticidade de acesso em sites HTTP. Protocolos:

- SSL: Secure Socket Layer
- TLS: Transport Layer Security

### SSL/TLS Handshake:



---

### Gerando o Certificado

A geração do certificado digital que será instalado em um domínio pode seguir os seguintes passos:

#### 1. Geração da Chave Privada

É comum utilizar o diretório /etc/ssl/certs para armazenar as chaves e certificados gerados.

O primeiro passo é gerar uma chave privada RSA:

```
# openssl genrsa -de3 -out dominio.key 1024
```



## 2. Geração do CSR (Certificate Signing Request)

Nessa passo geramos o arquivo CSR.

```
# openssl req -new -key dominio.key -out dominio.csr
```

Esse arquivo será enviado ao CA (Certification Authority), que responderá com o certificado (crt) que será instalado no servidor web.

## 3. Assinando o Próprio Certificado

Em ambientes de testes, em vez de utilizar um CA para a geração de um certificado, pode ser feito um procedimento interno, que simule um CA, assinando e gerando o próprio certificado.

Uma das formas de se fazer isso é pelo script /etc/pki/tls/misc/CA.pl, pelos seguintes passos:

```
# cd /etc/pki/CA
# /etc/pki/tls/misc/CA.pl -newca
# cp /etc/ssl/certs/dominio.csr newreq.pem
# /etc/pki/tls/misc/CA.pl -signreq
# cp newcert.pem /etc/ssl/certs/dominio.crt
```

---

## Instalando e Configurando o Certificado

As configurações SSL devem estar sempre dentro de um VirtualHost.

Os principais parâmetros são:

- Listen 443 – Habilitar o servidor a escutar a porta 443
- SSLEngine on - Habilita o recurso de SSL
- SSLCertificateFile – Arquivo do certificado digital para o domínio
- SSLCertificateKeyFile – Arquivo da chave privada utilizada para o certificado

Exemplo de Configuração:

```
<VirtualHost 192.168.1.220:443>
    ServerName www.dominioexemplo.com.br
    DocumentRoot /var/www/html
    Redirect /antigo.html /teste.php
    SSLEngine on
    SSLCertificateFile /etc/ssl/certs/dominioexemplo.com.br.crt
    SSLCertificateKeyFile /etc/ssl/certs/dominioexemplo.com.br.key
</VirtualHost>
```

---

## Certificado de Cliente

É possível exigir que o cliente possua um certificado específico para que possa acessar determinado conteúdo dentro do site.

Nesse caso o certificado deve ser gerado através dos seguintes passos:

### 1. Geração da Chave Privada

```
# openssl genrsa -de3 -out usuario.key 1024
```

### 2. Geração do CSR (Certificate Signing Request)

```
# openssl req -new -key usuario.key -out usuario.csr
```

### 3. Assinando o Próprio Certificado

```
# cd /etc/pki/CA
# cp /etc/ssl/certs/usuario.csr newreq.pem
# /etc/pki/tls/misc/CA.pl -signreq
# cp newcert.pem /etc/ssl/certs/usuario.crt
```

### 4. Gerar o Arquivo PKCS12, contendo a chave privada e o certificado:

```
# openssl pkcs12 -export -inkey usuario.key -in usuario.crt -out usuario.p12
```

### 5. Importar o arquivo usuario.p12 no navegador

### 6. Configurar os seguintes parâmetros no httpd.conf:

- SSLCACertificateFile – Indica o certificado do CA
- SSLCACertificatePath (opcional) – Diretório que contenha os certificados do CA local
- SSLVerifyClient require – Exige um certificado do cliente

Exemplo de Configuração:

```
<VirtualHost 192.168.1.220:443>
    ServerName www.dominioexemplo.com.br
    DocumentRoot /var/www/html
    SSLEngine on
    SSLCertificateFile /etc/ssl/certs/dominioexemplo.com.br.crt
    SSLCertificateKeyFile /etc/ssl/certs/dominioexemplo.com.br.key
    SSLCACertificateFile /etc/ssl/certs/cacert.pem
    <Directory /var/www/html/cert>
        SSLVerifyClient require
    </Directory>
</VirtualHost>
```

---

## Configurações de Segurança e SSL

O Apache possui uma série de configurações relacionadas tanto à segurança geral do servidor, como ao uso dos protocolos SSL/TLS.

As principais são:

- **SSLProtocol** – Define as versões suportadas para os protocolos SSL e TLS
- **SSLCipherSuite** – Define os ciphers (algoritmos) suportados no Handshake
- **SSLHonorCipherOrder** – Define se a ordem dos ciphers definido no SSLCipherSuite deve ser seguida
- **ServerTokens** – Define o nível de detalhes quanto à versão do Apache e seus módulos que pode ser disponibilizada. “Prod” ou “ProductOnly” é o menor (e recomendado) nível.
- **ServerSignature** – Define se será exibido um rodapé com informações do servidor em mensagens de erro.
- **TraceEnable** – Define se será possível utilizar o recurso do TRACE nas requisições

---

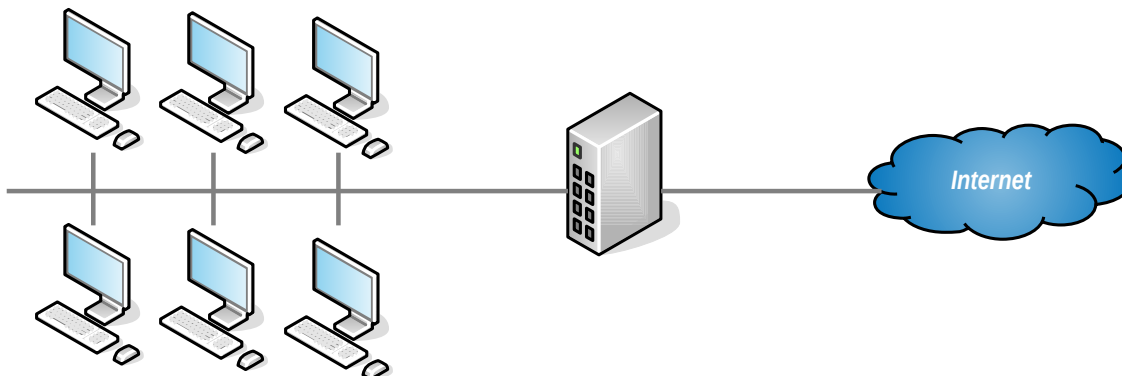
### **SNI (Server Name Indication)**

É uma extensão do SSL/TLS que permite o uso de VirtualHost em conexões HTTPS.

O recurso deve estar habilitado tanto no cliente (navegador) que irá enviar os dados da URL no Handshake SSL/TLS, quanto no servidor web.

## 208.3 – Implementando um Servidor Proxy (Squid)

Um proxy é um intermediário nas conexões entre os clientes e a Internet.



### Squid

O Squid é o principal servidor proxy para conexões HTTP e HTTPS, podendo também atuar com outros protocolos.

Arquivo de Configuração: **/etc/squid/squid.conf**

Arquivos de Logs: **/var/log/squid/**

Principais Comandos:

- **# squid -v** : Versão
- **# squid -k reconfigure** : Relê as configurações
- **# squid -z** : Recria os diretórios de cache

Configuração de Porta no squid.conf

- **http\_port 3128**

---

### Configurações de Cache

O seguinte parâmetro no squid.conf implementa o recurso de cache:

**cache\_dir ufs /var/spool/squid 100 16 256**

Nessa configuração temos que:

- **ufs** : Estrutura de armazenamento dos arquivos do cache
- **/var/spool/squid** : Diretório em que o cache será armazenado
- **100** : Quantidade máxima, em MB, de armazenamento
- **16** : Quantidade de diretórios no primeiro nível. Nomes em formato hexa
- **256** : Quantidade de subdiretórios em cada diretório. Nomes em formato hexa

## **Controle de Acesso**

Para definição das regras de controle de acesso são utilizadas acls, definidas no seguinte formato:

**acl <nome-da-acl> <tipo> <conteúdo>**

Os tipos possíveis de acl são:

- src : IP/Rede de Origem
- dst : IP/Rede de Destino
- port : Porta ou lista de portas
- srcdomain : Domínio de origem
- dstdomain : Domínio de destino
- time : Dia da Semana e Horário
  - Exemplo: time MTWHF 12:00-14:00
  - S = Domingo
  - M = Segunda
  - T = Terça
  - W = Quarta
  - H = Quinta
  - F = Sexta
  - A = Sábado
- proto : Protocolo utilizado
- browser : Navegador utilizado
- url\_regex : String ou expressão presente na URL
- urlpath\_regex : String ou expressão presente na URL, excluindo-se o endereço do domínio

Exemplos de acls:

```
acl localnet src 10.0.0.0/8
acl SSL_ports port 443
acl horario-comercial time MTWHF 09:00-18:00
acl entretenimento url_regex futebol esporte novela
acl proibidos url_regex -i "/etc/squid/proibidos.txt"
acl dominios dstdomain .facebook.com .globoesporte.com .windows.com
```

Após definidas as acls, elas devem ser utilizadas pelo parâmetro http\_access, no seguinte formato:

**http\_access allow/deny <acl>**

Exemplos de regras:

```
http_access deny entretenimento !horario-comercial
http_access deny proibidos
http_access allow dominios
http_access allow localnet
http_access deny all
```

## **Autenticação**

O Squid pode realizar a autenticação de usuário de várias formas, através de um simples arquivo com usuários e senhas, através de banco de dados, LDAP, e etc.

Na LPIC-2 pede-se apenas a configuração mais simples, através de arquivos texto gerados pelo htpasswd.

### **Definindo os Parâmetros da Autenticação**

# O realm define a mensagem a ser exibida

**auth\_param** basic realm Por favor identifique-se para conseguir acesso

# Define o modo de autenticação. Nesse caso também é indicado o arquivo que contém os usuários e senhas. Esse arquivo é criado pelo comando htpasswd, da mesma forma como foi criado para o controle de acesso no Apache.

**auth\_param** basic program /usr/lib64/squid/basic\_ncsa\_auth /etc/squid/passwords

### **Criando as Regras para Autenticação**

# Criação da acl do tipo proxy\_auth

**acl** autenticacao **proxy\_auth** **REQUIRED**

# Uso da acl

**http\_access** **allow** autenticacao

## 208.4 – Implementando o NGINX como WebServer e Proxy Reverso

O NGINX é o servidor web que mais cresce em uso no mercado principalmente devido à sua excelente performance em ambiente com muito tráfego.

Além de um Servidor Web completo, ele é muito utilizado como Proxy Reverso e para Balanceamento de Carga.

Principal Arquivo de Configuração: **/etc/nginx/nginx.conf**

Principais Comandos:

- `# nginx -t` : Verifica a sintaxe no arquivo de configuração
- `# nginx -s reload` : Recarrega as configurações

### Principais Configurações

O `nginx.conf` é estruturado através de blocos. As configurações referentes ao servidor HTTP são localizadas no **bloco `httpd { }`** e configurações específicas de cada servidor dentro do **bloco `server { }`**.

Os principais parâmetros de configuração são:

- `listen` : IP e Porta que o servidor escutará
- `server_name` : Nome de domínio ao qual o servidor pode responder
- `root` : Diretório que contém os arquivos que serão servidos
- `index` : Lista de arquivos considerados como Index

Exemplo de configuração:

```
http{
    ...
    server {
        listen      80;
        listen      [::]:80;
        server_name  www.dominioexemplo.com.br dominioexemplo.com.br;
        root        /var/www/html;
        index        index.html index.html index.php

        location / {
        }
    }
}
```

---

### Configuração do FastCGI para PHP

O FastCGI é um protocolo que faz uma interface entre o servidor web e programas ou scripts em PHP, C, Perl e etc.

Diferente do Apache que trabalha diretamente com o `mod_php`, o NGINX precisa de um processo a parte para servir os conteúdos desses programas. No caso do PHP o processo é o `php-fpm`, que roda na porta 9000 ou através de um socket.

Configurações utilizadas:

- **fastcgi\_pass** : Indica o IP/Porta ou Socket para o qual os requests PHP serão encaminhados
- **fastcgi\_param** : Determina parâmetros que serão enviados ao PHP

Exemplo de configuração:

```
location ~ \.php$ {
    fastcgi_pass 127.0.0.1:9000;
    fastcgi_param QUERY_STRING $query_string;
    fastcgi_param SCRIPT_FILENAME $document_root$fastcgi_script_name;
    include /etc/nginx/fastcgi_params;
}
```

---

## Configuração de Proxy Reverso

O NGINX pode ser configurado para encaminhar todo ou parte do tráfego para outro servidor, como um outro NGINX, Ou um Apache, Tomcat, Jboss e etc.

Para isso são utilizados os seguintes parâmetros:

- **proxy\_pass** : Determina o IP/Porta para o qual o request será encaminhado
- **proxy\_set\_header** : Adiciona e faz alterações no cabeçalho dos requests antes de encaminhá-los ao destino

Exemplo de configuração:

```
location / {
    proxy_pass http://192.168.1.210:8080
    proxy_set_header X-Real-IP $remote_addr;
    proxy_set_header X-Forwarded-For $remote_addr;
    proxy_set_header Host $host;
}
```



# Tópico 209 – Compartilhamento de Arquivos

## 209.1 – Configurações do Servidor SAMBA

### Processos e Configurações

O Samba funciona através de 2 processos principais:

- **smbd** – Responsável por possibilitar o compartilhamento de arquivos e impressoras através dos protocolos SMB ou CIFS
- **nmbd** – Processo que entende e pode responder requisições do protocolo NetBios, utilizado por sistemas Windows

Quanto às **portas** utilizadas temos:

- **smbd** – Portas TCP 139 e 445
- **nmbd** – Portas UDP 137 e 138

Há um terceiro processo utilizado quando o Samba está sendo inserido em um AD (Active Directory), que é o processo **winbind**. Esse processo TEM a função de fazer o Samba enxergar os usuários do AD.

---

### Arquivos de Configuração e Logs

O principal arquivo de **configuração** é o **/etc/samba/smb.conf**.

No arquivo smb.conf, a seção [Global] conterà os parâmetros de configuração geral do Samba. Dentre as centenas de opções, podemos citar:

- **workgroup** – Define o workgroup (grupo de trabalho) ao qual o Samba fará parte
- **netbios name** – Nome do servidor Samba no protocolo NetBios
- **realm** – Usado para identificar o domínio ao qual o Samba será ingressado
- **server string** – Identificação do Servidor
- **wins support** – Usado para habilitar o suporte ao WINS (Windows Internet Name Service)
- **server role** – Usado para definir a forma de funcionamento do Samba: O mais comum e o que é coberto pelo LPIC-2 é o “standalone server”.
- **unix password sync** – Utilizado para permitir que ao se alterar a senha de um usuário no samba, pelo comando smbpasswd, a senha do mesmo usuário no Unix/Linux também seja alterada
- **log file** – Caminho e nome do arquivo de log
- **username map** – Mapeamento de Usuários (explicado a seguir)
- **security** – Modo de Segurança (explicado a seguir)

Os **arquivos de log** estão presentes no diretório **/var/log/samba**.

---

### Compartilhamento de Diretórios

O compartilhamento de diretórios no Samba é feito através da definição de uma [tag] com o nome do compartilhamento e uma série de parâmetros. Os principais parâmetros são:

- comment = Descrição do Compartilhamento
- path = Caminho do diretório a ser compartilhado
- guest ok = Permitir ou não o acesso ao compartilhamento de usuários convidados, ou seja, usuários não cadastrados no samba
- read only = Define se o acesso será apenas para leitura
- writeable = É o oposto do “read only”, define que o acesso pode ser feito com permissão de escrita
- browseable = Define se o compartilhamento será visível no servidor
- valid users = Define os usuários que poderão ter acesso ao compartilhamento

A tag **[Homes]** é utilizada para definir um compartilhamento padrão para todos os usuários do sistema.

Abaixo alguns exemplos de compartilhamento de diretórios.

[Homes]

```
comment = Home Directories
browseable = no
read only = no
```

[DiretorioX]

```
comment = Compartilhamento do Diretório X
path /opt/diretorioX
browseable = yes
writeable = yes
guest ok = no
```

Um ponto importante é que sistemas Windows identificam qualquer compartilhamento cujo nome termine com o símbolo \$, como um compartilhamento oculto, os chamados “hidden shares”. Por exemplo:

[Secreto\$]

```
comment = Diretório Oculto
path = /tmp
guest ok = no
```

---

## **Compartilhamento de Impressoras**

O Samba também pode ser utilizado para compartilhar com a rede impressoras instaladas no sistema. Para isso basta criar o seguinte compartilhamento:

**[Printers]**

```
comment = Impressoras
path = /var/spool/samba
browseable = yes
printable = yes
```

Também é importante criar o compartilhamento que disponibilizará os drives das impressoras instaladas no servidor, para que estes possam ser baixados e instalados pelas estações clientes (Windows):

#### **[print\$]**

```
comment = Drivers das Impressoras
path = /var/lib/samba/printers
browseable = yes
read only = yes
guest ok = no
```

---

### **Usuários no Samba**

O Samba não utiliza os mesmos usuários configurados no Unix/Linux, contidos nos arquivos passwd e shadow. É preciso criar os usuários no Samba.

Para isso é utilizado o comando **smbpasswd**. As principais opções são:

- -a : adiciona um novo usuário
  - -x : deleta um usuário
  - -d : desabilita um usuário
  - -e : habilita um usuário
  - <sem opções> : Altera a senha de um usuário existente
- 

### **Montando Remotamente os Compartilhamentos do Samba**

O modo mais usado para montar no Linux os compartilhamentos criados no Samba é através do protocolo CIFS (Common Internet File System).

Com ele instalado, através do pacote cifs-utils, basta montar o compartilhamento através do comando **mount** ou pelo **/etc/fstab**.

Exemplo de uso do mount:

```
# mount -t cifs -o username=usuario-samba,password=senha //IP/Compartilhamento /mnt/diretorio
```

Outra opção é pela configuração do fstab. Alguns exemplos

```
//IP/Compartilhamento /mnt/diretorio cifs username=usuario-samba,password=senha 0 0
//IP/Compartilhamento /mnt/diretorio cifs credentials=arquivo 0 0
```

\* Utilizando a opção credentials, o “arquivo” conterá as informações de usuário e senha.

---

### **Mapeamento de Usuários**

Há no Samba uma opção chamada “**username map**” que precisa ser utilizada principalmente quando o nome do usuário no Windows é diferente do nome do usuário no Samba. Também é útil para fazer com que o Samba entenda vários usuários como um só.

A configuração é feita da seguinte maneira no arquivo smb.conf:  
**username map** = /etc/samba/nome-do-arquivo

O arquivo indicado conterá uma linha para cada referência:  
<usuário do samba> = <usuário informado no acesso>

---

## Níveis de Segurança

Através do parâmetro “**security**” no smb.conf, é possível definir o nível ou tipo de segurança utilizado pela Samba.

São dois tipos macro: “user level” e “share level”, conforme descrito abaixo:

- **Share Level**
    - **share** = Cada compartilhamento possui uma senha para acesso
  - **User Level**
    - **user** (padrão) – O acesso aos compartilhamento é feito através da autenticação de usuários, com o uso de senhas individuais
    - **server** – A autenticação é feita por um servidor remoto (Não recomendado por falhas de segurança)
    - **domain** – O Samba irá validar o usuário e senha em um Controlador de Domínio Windows
    - **ads** – O Samba irá ingressar como membro de um AD (Active Directory)
- 

## Samba como Membro do AD

Primeiramente, para que o Samba possa ingressar em um AD, é preciso que o serviço **winbind** esteja disponível. É através deste serviço que o Samba conseguirá enxergar os usuários cadastrados no AD.

Referente às configurações, os seguintes parâmetros devem estar devidamente definidos:

- **workgroup** = Grupo de trabalho utilizado. Exemplo: DOMINIOEXEMPLO
- **realm** = Domínio utilizado no Windows. Exemplo: dominioexemplo.com.br
- **security** = ads
- **netbios name** = Nome do Samba no NetBios. Exemplo: servidor-samba

Outros pontos importantes:

- Configura o servidor em que o Samba está instalado para utilizar o servidor DNS do Windows
- Configurar o arquivo /etc/nsswitch.conf, incluindo a referência ao winbind ao final dos registros passwd e shadow.

Por fim, para fazer o Samba ingressar no AD é utilizado o comando “net”:  
# net ads join -U Administrator

Para testar, basta utilizar o comando wbinfo:  
# wbinfo -u : Para listar os usuários

# wbinfo --ping-dc : Para realizar um teste de acesso ao DC (Domain Controller)

---

## **Principais Comandos utilizados no Samba**

### **Comando smbpasswd**

Função: Criar e gerenciar os usuários do Samba.

#### Uso e Principais Opções:

- -a : adiciona um novo usuário
- -x : deleta um usuário
- -d : desabilita um usuário
- -e : habilita um usuário
- <sem opções> : Altera a senha de um usuário existente

### **Comando smbclient**

Função: Permite o acesso a compartilhamentos feitos através dos protocolos SMB/CIFS, incluindo no Samba.

#### Uso e Principais Opções:

# smbclient //IP/Compartilhamento -U usuario  
# smbclient -L //IP -U usuario : Lista os compartilhamentos disponíveis

### **Comando testparm**

Função: Verifica a sintaxe das configurações do arquivo smb.conf, exibindo também os parâmetros definidos.

#### Uso e Principais Opções:

# testparm : Verifica a sintaxe e exibe os parâmetros definidos no arquivo  
# testparm -v : Verifica a sintaxe e exibe todos os parâmetros do Samba, incluindo os parâmetros default.

### **Comando smbstatus**

Função: Exibe detalhes das atuais conexões no Samba

#### Uso e Principais Opções:

# smbstatus : Exibe os processos e compartilhamentos  
# smbstatus -S : Exibe apenas os compartilhamentos em uso

## **Comando nmblookup**

Função: Faz a resolução de nomes NetBios em IP e vice versa

Uso e Principais Opções:

```
# nmblookup -S servidor-samba : Verifica o status do host pelo nome NetBios
# nmblookup -A 192.168.1.210 : Verifica o status do host pelo IP
```

## **Comando samba-tool**

Função: Ferramenta de administração do Samba. Muito utilizado quando o Samba é usado como Controlador de Domínios (PDC).

Uso e Principais Opções:

```
# samba-tool processes : Lista os processos
# samba-tool dbcheck : Verifica a base de dados do AD local
```

## **Comando smbcontrol**

Função: Envia sinais e comandos aos processos do samba: smbd, nmbd, winbind

Uso e Principais Opções:

```
# smbcontroll all reload-config
# smbcontrol winbind shutdown
```

## **Comando net**

Função: Ferramenta de administração similar ao comando net do Windows.

Uso e Principais Opções:

```
# net -S localhost -U usuario share
# net -S servidor-samba time
```

## 209.2 – Configurações do Servidor NFS

O NFS (Network File System) permite que diretórios compartilhados remotamente sejam montados como locais.

Nesse modelo, o **servidor** NFS é quem está disponibilizando um diretório para ser compartilhado e o **cliente** realiza a montagem desse compartilhamento remotamente.

A versão atual do NFS é a 4.1, mas o software de cliente/servidor suporta também todas as versões anteriores. A versão 3.0 ainda é muito utilizada.

### Os Processos do NFS

Um servidor NFS funciona através de vários processos, entre eles temos 3 principais:

- **rpcbind** (portmap) : Realiza a função de mapeamento de portas (portmapper), ou seja, permite ao cliente identificar as portas em que está sendo disponibilizada o serviço. Essas portas são por padrão dinâmicas, então o serviço é essencial.
- **nfsd** : Processo que atende as requisições do cliente.
- **rpc.mountd** : Executa as solicitações encaminhadas pelo nfsd, montando efetivamente os compartilhamentos entre cliente-servidor.

---

### Criando os Compartilhamentos

No servidor, os compartilhamentos devem ser criados no arquivo **/etc/exports** (mais comum), ou em arquivos dentro do diretório **/etc/exports.d/**.

Nesse arquivo, cada linha define um compartilhamento, no seguinte formato:

<Diretorio-Compartilhado> <Clientes-Permitidos>(Opções)

Dentre as opções informadas, as mais comuns e importantes são:

- **rw** – Permite Leitura e Escrita.
- **ro** – Permite Apenas Leitura (read-only). Opção padrão.
- **sync** – Operações síncronas. Opção padrão.
- **async** – Operações assíncronas, utilizando um cache interno.
- **no\_subtree\_check** – O subtree\_check é feito quando um subdiretório é compartilhado mas a partição em que o diretório está não. Nesses casos o NFS faz uma checagem para verificar se o subdiretório continua na mesma partição. O no\_subtree\_check desabilita essa verificação. É a opção padrão.
- **root\_squash** – Acessos como root no cliente são feitos com usuário anônimo (nobody) no servidor. Opção padrão.
- **no\_root\_squash** – Acessos como root no cliente são feitos como root no servidor.
- **all\_squash** – Acessos de usuários comuns no cliente são feitos com usuário anônimo (nobody) no servidor.
- **no\_all\_squash** – Acessos de usuários comuns no cliente são feitos com o mesmo usuário (quando existir) no servidor. Opção padrão.

Por exemplo:

```
/var/log      192.168.1.111(ro,sync,no_subtree_check)
/opt/nfs      192.168.1.0/24(rw,async,no_subtree_check)
```

## O comando **exportfs**

Após configurar os compartilhamentos, deve ser utilizado o comando “**exportfs**” para disponibilizá-los aos clientes. As principais opções do **exportfs** são:

- -a : Habilita todos os compartilhamentos definidos no `/etc/exports`
- -r : Reexporta os compartilhamentos, após mudanças nas configurações por exemplo
- -f : Faz um refresh em todas as conexões do NFS
- -ua : Desabilita todos os compartilhamentos
- -v : Exibe os compartilhamentos disponíveis com detalhes
- <sem opções> : Exibe os compartilhamentos disponíveis

O **exportfs** também pode ser utilizado para exportar um diretório temporariamente, utilizando-se o seguinte formato:

```
# exportfs <IP-Cliente>:<compartilhamento> -o <lista-de-opções>
```

Por exemplo:

```
# exportfs 192.168.1.220:/home/ricardo -o ro,async
```

---

## Montando um Compartilhamento Remotamente

Os compartilhamentos são montados diretamente com o uso do comando **mount**, mais precisamente o **mount.nfs**, ou através do **/etc/fstab**.

A sintaxe do comando **mount** é:

```
# mount -t nfs <IP-Servidor>:<Compartilhamento> <Ponto-de-Montagem>
```

Por exemplo:

```
# mount -t nfs 192.168.1.210:/var/log /mnt/log-remoto
```

A definição no arquivo `/etc/fstab` segue o exemplo abaixo:

```
192.168.1.210:/var/log /mnt/log-remoto nfs defaults 0 0
```

---

## O comando **showmount**

Função: O **showmount** é usado para listar os compartilhamentos disponíveis, tanto no servidor quanto a partir do cliente.

Uso e Principais Opções:

No servidor, basta utilizar:

```
# showmount -e
```



No cliente, o uso é:  
# showmount -e IP-Servidor

---

### **O comando nfsstat**

Função: Exibir estatísticas do NFS, tanto no cliente como no servidor.

#### Uso e Principais Opções:

As principais opções são:

- -m : Exibe estatísticas dos compartilhamento atualmente montados
  - -c : Estatísticas do cliente NFS
  - -s : Estatísticas do servidor NFS
  - -3 : Estatísticas dos compartilhamentos que usam a versão 3 do NFS
  - -4 : Estatísticas dos compartilhamentos que usam a versão 4 do NFS
  - -l : Exibe as estatísticas em um formato de lista
- 

### **O comando rpcinfo**

Função: Exibir informações do protocolo RCP (Remote Procedure Call), utilizado pelo NFS.

#### Uso e Principais Opções:

A principal opção é o “**rpcinfo -p**”, que exibe a lista de processos e portas utilizados no momento pelo NFS.

---

### **O Pseudo FileSystem**

A partir da versão 4, ao invés do NFS exportar cada compartilhamento individualmente, todos os compartilhamentos são exportados como se fossem um filesystem único, o que é chamado de pseudo-filesystem.

Isso é feito internamente pelo NFS e é transparente para o cliente.

---

### **TCP Wrapper**

Assim como outros serviços do Linux, o NFS também utiliza as bibliotecas do TCP Wrapper, dessa forma é possível utilizar os arquivos /etc/hosts.allow e /etc/hosts.deny para adicionar uma camada extra de segurança no uso do serviço.

Por exemplo, para liberar apenas o acesso da rede 192.168.1.0/24 ao servidor, pode ser feita a seguinte configuração:

No /etc/hosts.allow:  
rpcbind: 192.168.1.0/24

No /etc/hosts.deny:  
rpcbind: ALL

\* Em alguns sistemas, ao invés de rpcbind o processo pode ter o nome portmap.

# Tópico 210 – Administração dos Clientes de Rede

## 210.1 – Configurações de DHCP

### O serviço DHCP

Processo: dhcpd

Portas: UDP 67 e 68

---

### Configuração

Arquivo de Configuração Principal: /etc/dhcp/dhcpd.conf

Principais Configurações:

- default-lease-time = Tempo de verificação se o IP alocado ainda está em uso.
- max-lease-time = Tempo máximo de concessão do IP. Após esse tempo o IP será liberado, caso a máquina ainda esteja ativa, o IP será realocado para a mesma máquina.
- log-facility = Define o nome da facility que poderá ser utilizada como referência no syslog.
- option domain-name = Define o domínio a ser informado.
- option domain-name-servers = Servidores DNS que podem ser utilizados pelo cliente
- subnet = Define a rede e o range de IPs que será concedido. Exemplo:  
    subnet 192.168.1.0 netmask 255.255.255.0 {  
        range 192.168.1.100 192.168.1.200;  
        option routers 192.168.1.1; <=== Gateway a ser utilizado pelo cliente  
    }
- host = Usado para definir um IP específico a um hardware específico. Exemplo:  
    host exemplo {  
        hardware ethernet 08:00:27:0d:73:cc; <=== MAC Address  
        fixed-address 192.168.1.50; <=== IP Fixo  
    }
- deny unknown-clients = Só distribui IPs para Hosts conhecidos
- allow bootp = Habilita o servidor a responder a requisições bootp.
  - Exemplo de configuração para bootp  
        host exemplo {  
            hardware ethernet 08:00:27:0d:73:cc; <=== MAC Address  
            fixed-address 192.168.1.50; <=== IP Fixo  
            filename “vmlinux.exemplo”;  
            server-name “boot.dominio.com”;  
        }

Arquivo com Registro de Atribuições: /var/lib/dhcp/dhcpd.leases

---

### Logs

Os logs são normalmente gerados nos arquivos **/var/log/messages** ou **/var/log/syslog**.

No entanto, através da opção de configuração “**log-facility**” no `dhcpd.conf`, pode também ser configurado através do `rsyslog` para que outro arquivo seja utilizado.

Os logs também podem ser acessados pelo **journalctl**:

```
# journalctl -e -u dhcpd
```

---

## **DHCP Relay**

O DHCP relay é um processo que redireciona as requisições DHCP da rede local para um servidor DHCP central, normalmente em outra rede.

O processo responsável é o **dhcrelay**. Por exemplo:

```
# dhcrelay -i eth0 servidor.empresa.com.br
```

---

## **IPv6 – radvd**

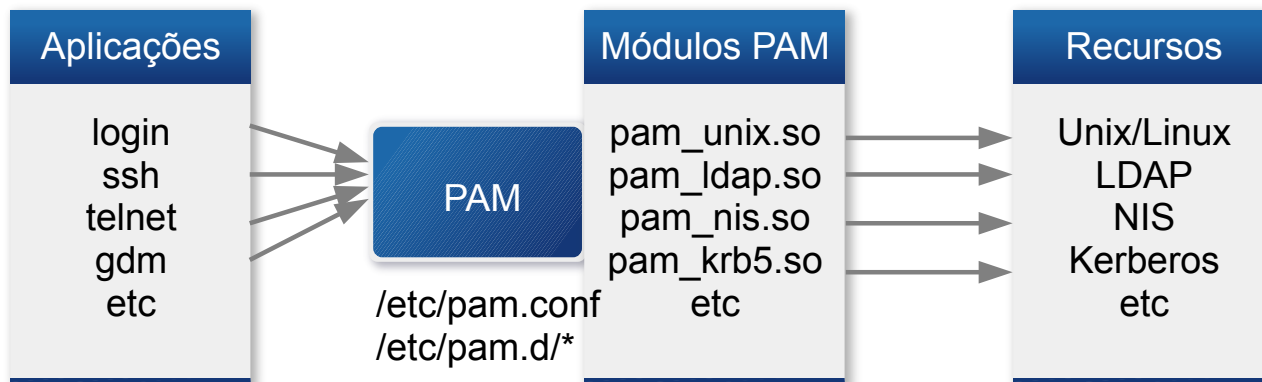
Em redes IPv6, o processo **radvd** é equivalente ao serviço DHCP, pois distribui aos clientes as informações da rede local e do gateway a ser utilizado,, serviço chamado de “router advertisement”, possibilitando então a autoconfiguração da máquina. O processo é parte do protocolo NDP (Neighbor Discovery Protocol).

O arquivo de configuração é o **/etc/radvd.conf**.

## 210.2 – Autenticação via PAM

PAM = Pluggable Authentication Modules.

É uma interface para autenticação de usuários em diversas aplicações.



### Configurações

As configurações ficam em arquivos específicos no diretório **/etc/pam.d/** ou no **/etc/pam.conf**. O mais comum é que cada serviço possua sua própria configuração no **/etc/pam.d**. Se houver duplicidade de configurações, a configuração do **pam.conf** é ignorada.

As configurações são basadas em 4 parâmetros:

- Tipo
- Controle
- Módulo
- Argumentos do Módulo (opcional)

#### Tipo:

- **auth** : Verifica a autenticidade do usuário. Normalmente por usuário e senha, mas também por chip, biometria e etc.
- **account** : Verifica se o usuário pode usar o serviço, se não há nenhum bloqueio de acesso.
- **password** : Definições referentes à atualização da autenticação.
- **session**: Algum procedimento que deve ser realizado após o login, antes do usuário receber o acesso.

#### Controle:

- **requisite** : Se o módulo falhar, todo o processo é interrompido.
- **required** : Se o módulo falhar, o acesso é negado, mas os demais módulos serão invocados.
- **sufficient** : Se o módulo tiver sucesso, a falha de outros módulos serão ignoradas. Uma falha não é fatal.
- **optional** : Sucesso ou falha não é relevante, a menos que seja o único.

#### Principais Módulos:

- **pam\_unix.so** – Relacionado principalmente ao passwd/shadow
- **pam\_limits.so** – Limitação de Recursos
- **pam\_ldap.so** – Acessos via LDAP

- **pam\_cracklib.so** – Checagem de senhas fracas
- **pam\_listfile.so** – Uso de arquivos externos para controle
- **pam\_sss.so** – Uso do SSS
- **pam\_krb5.so** – Uso do Kerberos 5 para Autenticação
- **pam\_userdb.so** – Uso de Datafiles .db
- **pam\_nologin.so** – Uso do /etc/nologin
- **pam\_time.so** – Recursos de controle por horário
- **pam\_console.so** – Controle de acesso ao console por usuário

#### Exemplos:

```
/etc/pam.d/login
auth    requisite pam_nologin.so
session required pam_limits.so
```

Para realizar a mesma configuração no /etc/pam.conf, é preciso incluir no primeiro campo o nome do serviço, por exemplo:

```
login auth    requisite    pam_nologin.so
login session required    pam_limits.so
```

### **LDAP com PAM**

Autenticação via LDAP é feito através do módulo **pam\_ldap.so**

Exemplo de configuração do /etc/pam.d/login:

```
auth    sufficient    pam_ldap.so
auth    required      pam_unix.so  try_first_pass
account sufficient    pam_ldap.so
account required      pam_unix.so
```

### **SSSD (System Security Services Daemon)**

Interface criada como melhoria ao PAM. O principal objetivo é ser uma interface para obter informações dos usuários a partir de diversas fontes, principalmente através do PAM e o do NSS e muito usado para comunicação com o LDAP e com serviços AD do Windows.

O processo é o sssd e o arquivo de configuração é o /etc/sss.conf.

## 210.4 – Configurando um Servidor OpenLDAP

\* Propositadamente alternado com o sub-tópico 210.3

### Definições e Conceitos

LDAP: Lightweight Directory Access Protocol

- É um protocolo utilizado para armazenamento e acesso a uma base de dados no **modelo de árvore**.
- Muito utilizado para armazenar informações de usuários, funcionários, equipamentos e etc.
- Modelo favorece a performance na leitura de dados.
- Cada nó representa um conjunto de atributos e valores.
- Cada nó utiliza uma tipo de identificador, os mais utilizados são:
  - DC = Domain Component
  - OU = Organizational Unit
  - CN = Common Name
- O **DN** (Distinguished Name) é um identificador único de cada nó e é composto pela identificação do caminho até este nó, por exemplo:
  - dn: cn=Ricardo,ou=suporte,dc=empresa,dc=com

A cada nó podem ser atribuídos uma série de atributos e valores, esses atributos fazem parte de um objectClass e os objectClasses estão agrupados em Schemas:

- Schemas
  - ObjectClasses
    - Attributes

---

### O Servidor LDAP

O OpenLDAP é a implementação de LDAP mais utilizada em ambientes Linux.

O processo é o **slapd**, que utiliza a porta TCP 389 por padrão e a porta 636 para LDAPS (LDAP Seguro).

As configurações são realizadas no diretório **/etc/ldap** ou **/etc/openldap**.

Por padrão, os bancos de dados são armazenados no diretório **/var/lib/ldap**.

---

### Configurações

As configurações do OpenLDAP podem ser feitas através do arquivo **/etc/ldap/slapd.conf** ou através das definições do diretório **/etc/ldap/slapd.d/**

As principais opções do arquivo slapd.conf são:

- **loglevel** = Nível de log.
- **backend** = Modelo de armazenamento dos dados, por exemplo hdb, bdb, entre outros.
- **database** = Define o início da definição de um banco de dados e deve ser seguido pelo tipo de backend (hdb,bdb,etc)

- suffix = Endereço base do banco de dados. Por exemplo: dc=empresa,dc=com
  - rootdn = Definição do administrador do BD
  - rootpw = Senha do administrador do BD
  - directory = Diretório que armazenará os dados
- 

## **LDIF - LDAP Data Interchange Format**

Formato de arquivo utilizado para importar, exportar e modificar dados de bancos LDAP.

As alterações são realizadas através das seguintes operações, também chamados de **changetype**:

- modify = Modifica os atributos de um DN
  - add = Adiciona um atributo
  - delete = Remove um atributo
  - replace = Altera o valor de um atributo
- add = Adiciona um novo DN
- delete = Remove um DN existente
- modrdn = Altera a descrição de um DN

Exemplo:

```
# cat altera-eduardo.ldif
```

```
dn: cn=Eduardo,ou=funcionarios,ou=testes,dc=dominioexemplo,dc=com,dc=br
```

```
changetype: modify
```

```
replace: mail
```

```
mail: silva.eduardo@dominioexemplo.com.br
```

---

## **Principais Comandos**

- slapcat – Exibe todos os registros do LDAP ou de bases específicas. Pode ser usado para exportar dados de um banco em um arquivo no formato LDIF.
- slappasswd – Utilizado para gerar uma senha criptografada para ser utilizada em arquivos de configuração do sistema.
- slapindex – Utilizado para reindexar os registros de uma base LDAP
- slapadd – Adiciona registros diretamente à base LDAP. Para ser utilizado o serviço precisa estar parado.
- slaptest – Verifica a sintaxe das configurações do servidor LDAP.



## 210.3 – Uso do Cliente LDAP

Para o acesso e alteração de dados de uma base LDAP, são utilizados comandos que enviam as requisições através do protocolo LDAP pela rede ou localmente.

Os principais comandos e alguns exemplos de uso são:

- ldapsearch - Realizar consultas.
  - # ldapsearch -x -h localhost -b "dc=dominioexemplo,dc=com" cn=Eduardo
  - # ldapsearch -x -h localhost -b "dc=dominioexemplo,dc=com" '(&(cn=Carlos)(sn=Almeida)))'
  - # ldapsearch -x -h localhost -b "dc=dominioexemplo,dc=com" sn=A\*
- ldapadd – Adicionar registros.
  - # ldapadd -x -W -D "cn=admin,dc=dominioexemplo,dc=com" -f novo-funcionario.ldif
- ldapdelete – Remover registros.
  - # ldapdelete -h localhost -D "cn=admin,dc=dominioexemplo,dc=com" -W "cn=Eduardo,ou=funcionarios,ou=testes,dc=dominioexemplo,dc=com"
  - # ldpelete -h localhost -D "cn=admin,dc=dominioexemplo,dc=com" -W -f arquivo.ldif
- ldapmodify – Alterar registros e seus atributos
  - # ldapmodify -x -D "cn=admin,dc=dominioexemplo,dc=com" -f arquivo.ldif
- ldappasswd – Alterar o atributo de senha de um registro de usuário
  - # ldappasswd -x -h localhost -D "cn=admin,dc=dominioexemplo,dc=com" -S -W uid=ricardo,ou=funcionarios,ou=suporte,dc=dominioexemplo,dc=com

# Tópico 211 – Serviços de E-mail

## 211.1 – Utilizando Servidores de E-mail

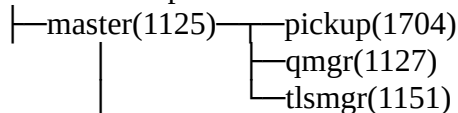
### Conceitos sobre SMTP

- SMTP = Simple Mail Transfer Protocol
- MTA = Mail Transfer Agent
- MTA = Servidor SMTP = Servidor de E-mail
- Portas:
  - SMTP = TCP 25
  - SMTPS = TCP 465

---

### Postfix

- Um dos principais MTAs do mercado
- O postfix funciona com um processo principal, chamado “**master**”, que invoca vários sub-processos. Por exemplo:



- Logs: /var/log/mail\*
- Arquivos de Configuração: /etc/postfix
  - **master.cf** – Configuração do processo master
  - **main.cf** – Configurações do serviço SMTP
- Filas Internas do Postfix: /var/spool/postfix
- Filas de E-mail (Mailbox): **/var/mail ou /var/spool/mail**

---

### Principais Configurações do Postfix

O principal arquivo de configurações do postfix é o main.cf. Abaixo os principais parâmetros:

- myorigin – Define o domínio/hostname que será indicado como origem dos e-mails, quando este não for informado.
- mydestination – Lista de domínios que o servidor aceita para entrega local de e-mails.
- mynetworks – IPs/Redes que podem utilizar o servidor para envio de e-mails a domínios externos. Essa permissão e envio de e-mails é também chamada de Relay.
- relay\_domains – Para quais domínios serão liberados o envio de e-mails, mesmo que o IP/Rede não esteja liberado para Relay.
- inet\_interfaces – Interfaces em que o serviço SMTP estará disponível.
- home\_mailbox – Define se o SMTP utilizar Mailbox ou Maildir para armazenamento local das mensagens.
- mailbox\_command - Indica o software responsável pela entrega local das mensagens.
- canonical\_maps (sender , canonical ) - Utilizado para alterar o remetente, destinatário ou ambos.
- smtpd\_use\_tls = yes – Habilita o uso do TLS.

- smtpd\_tls\_security\_level = (none/encrypt/may) – Define se o servidor irá ou não utilizar TLS. “may” disponibiliza como uma opção, “encrypt” como obrigatório.

\* Dentro do main.cf, é possível configurar um parâmetro em mais de uma linha, desde que a linha seguinte comece com um espaço em branco.

---

## **Alias**

O arquivo **/etc/aliases** pode ser utilizado para relacionar e-mails e usuários. Por exemplo:

suporte: ricardo  
admin: ricardo

Esta configuração define que todos os e-mails enviados para suporte@dominio e admin@dominio serão enviados ao usuário ricardo.

Após realizar essa configuração, o comando **newaliases** deve ser utilizado.

---

## **Comandos**

Ao utilizar o postfix o administrador tem disponível uma série de comandos específicos do postfix, além de alguns utilizados para compatibilidade com o sendmail.

Os principais comandos são:

- mailq – Verificar a fila de e-mails do sistema. A mesma informação pode ser verificada pelos comandos “sendmail -bp” e “postqueue -p”
- postfix – Operações de gerenciamento do servidor postfix.
- postconf – Visualização das configurações do postfix
- postcat – Visualização das mensagens dentro da fila do postfix (/var/spool/postfix)
- postqueue – Visualização e manipulação das mensagens em fila

---

## **Sendmail**

Configuração em: /etc/mail

- **/etc/mail/sendmail.cf** – Arquivo de configuração principal utilizado pelo servidor
- **/etc/mail/\*.mc** – Arquivos de configuração macro. Usados como origem ao sendmail.cf.

---

## **Exim**

Configuração em: /etc/exim/exim.conf ou /etc/exim4/exim4.conf

## 211.2 – Gerenciando a Entrega de E-mails

### Sieve

Linguagem de Programação utilizada na criação de filtros de mensagens no lado do servidor. O Sieve é disponibilizado como um recurso de MDAs (Mail Delivery Agent) como o Dovecot.

- **Tipos de Ação:**
  - keep: Apenas grava a mensagem na mailbox
  - fileinto: Grava uma cópia da mensagem em algum diretório
  - redirect: Reencaminha a mensagem a outro e-mail
  - discard: Descarta a mensagem sem aviso
  - reject: Recusa a mensagem retornando uma aviso
- **Controle:**
  - require: Adiciona suporte a extensões externas
  - stop: Pára o processamento do script
  - if: Implementa verificações condicionais
- **O que pode ser verificado?**
  - address
  - envelope
  - body
  - subject
  - size
  - header
- **Condições de Testes**
  - is – Busca por uma string exata
  - contains – Busca por parte de uma string
  - match – Uso de \* e ?
  - regex – Uso de Expressões Regulares
  - allop (e)
  - anyof (ou)
  - exists

Exemplo:

```
require ["fileinto"];
if header :contains "subject" ["curriculo", "cv"]
{
    fileinto "cvs";
    redirect "rh@dominio.com.br";
}
```

---

### Dovecot Vacation Extension

Regra utilizada para implementar auto-resposta de férias:

```
require ["vacation"];
```

```
vacation
: days 1
```

: subject “Assunto da Auto-Resposta”  
: addresses [“[email@dominio](#)”, “[email2@dominio](#)”]  
“Mensagem que será retornada ao remetente”;

Sendo que:

- days – Limita o número de auto-respostas para o mesmo remetente em determinado período
- subject – Assunto do e-mail da auto-resposta
- addresses – Possíveis e-mails relacionados aos usuários

---

## **Procmail**

Alternativa ao Dovecot/Sieve para filtro de mensagens.

Regras gerais em: /etc/procmailrc

Regras individuais em: ~/.procmailrc

As regras são compostas de:

- Início da Regra
- Condições
- Ação

Por exemplo:

```
:0  
* ^From.*usuario@.*  
diretorio
```

---

## **Mailbox vs Maildir**

Formatos de armazenamento das mensagens nas contas locais dos usuários.

- Mailbox
  - Formato padrão
  - Mensagens armazenadas no diretório /var/spool/mail/
  - Cada usuário possui um arquivo único, com todas as mensagens
- Maildir
  - As mensagens são armazenadas no diretório ~/Maildir/
  - Dentro do diretório Maildir existem 3 sub-diretórios: new, cur, tmp
  - Cada e-mail é um arquivo

No postfix a configuração é feita no main.cf, no parâmetro home\_mailbox.

No dovecot, a configuração é feita no arquivo /etc/dovecot/conf.d/10-mail.conf, no parâmetro mail\_location.

## 211.3 – Gerenciando a Entrega Remota de E-mails

### Conceitos

A entrega remota de e-mails é realizada através dos protocolos IMAP e POP3, que são disponibilizados através de MDAs como o Dovecot.

- **POP3 – Post Office Protocol**
  - Porta 110 / 995 (pop3s)
  - Download dos E-mails (deletando do servidor)
  - O e-mail pode ser lido offline
  - Não faz sincronização entre diferentes dispositivos
- **IMAP – Internet Message Access Protocol**
  - Porta 143 / 993 (imaps)
  - Ver e Acessar os e-mails, sem deletar do servidor
  - É preciso estar online para acessar os e-mails
  - Recomendado quando o acesso é feito por múltiplos devices
  - Faz a sincronização dos diretórios nos diferentes dispositivos

---

### Principais Configurações do Dovecot

As configurações do Dovecot estão distribuídas em diversos arquivos no diretório /etc/dovecot e /etc/dovecot/conf.d.

Abaixo as principais configurações:

- auth\_mechanisms – Define as opções de autenticação disponíveis
- inet\_listener (port) – Define as portas utilizadas
- protocols – Protocolos habilitados, em geral imap e pop3
- mail\_location – Define se é utilizado o padrão Mailbox ou Maildir
- ssl – Define se será utilizado o SSL
- verbose-proctitle – Mostra mais informações das conexões na descrição do processo, visto pelo comando ps, por exemplo o nome do usuário e IP.

---

### Comandos do Dovecot

Os principais comandos são:

- doveconf – Exibe um dump das configurações ativas do dovecot
- doveadm – Diversas opções referentes ao gerenciamento do Dovecot
  - who – Usuários logados no momento
  - auth test – Teste de autenticação de um usuário
  - reload – Relê as configurações
  - user – Verifica informações de determinado usuário

---

### Courier

Alternativa ao Dovecot para prover o IMAP e POP3.

Configurações em **/etc/courier/imapd** e **/etc/courier/pop3d**.

Principais configurações:

- **PORT** – Definição da porta do serviço
- **MAXDAEMONS** – Número máximo de servidores iniciados
- **MAXPERIP** – Número máximo de conexões a partir do mesmo IP
- **ADDRESS** – Definir os endereços que irão ouvir conexões no serviço

# Tópico 212 – Segurança do Sistema

## 212.1 – Configurando um Roteador

### Classes de IPs / IPs Privados

Classe	Primeiro Octeto	Range	IPs Privados
A	1-126	1.0.0.0 – 126.255.255.255	10.0.0.0 – 10.255.255.255
B	128-191	128.0.0.0 – 191.255.255.255	172.16.0.0 – 172.31.255.255
C	192-223	192.0.0.0 – 223.255.255.255	192.168.0.0 – 192.168.255.255

---

### Configurando um Roteador

Para habilitar o roteamento entre redes dentro de um servidor Linux, os seguintes parâmetros devem ser habilitados:

- IPv4: /proc/sys/net/ipv4/ip\_forward (1)
- IPv6: /proc/sys/net/ipv6/conf/all/forwarding (1)

Internamente as rotas podem ser gerenciadas de forma estática ou dinâmica.

Para rotas estáticas, é utilizado principalmente o comando “route”. Por exemplo:

- # route -n
- # route add -net 172.16.32.0/24 gw 192.168.1.100:80

Para rotas dinâmicas, utilizados em roteadores que lidam com muitas redes e conexões, são usados alguns serviços como:

- routed
- gated
- quagga
- bird

---

### Iptables

O iptables é a aplicação utilizada para gerenciar o recurso netfilter do kernel do Linux.

Basicamente, ele é administrado através de 3 **tabelas**:

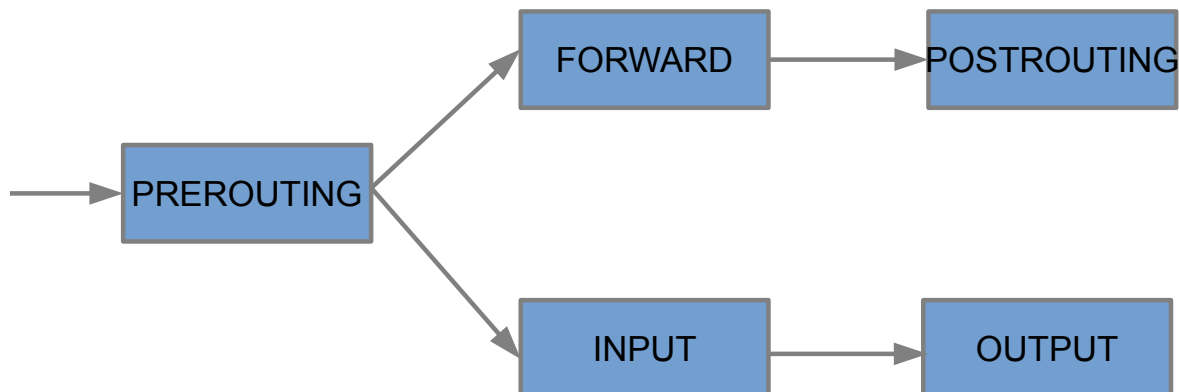
- **filter**: Tabela padrão do iptables. Utilizada para criar filtros de pacotes, bloqueando e/ou liberando o tráfego.
- **nat**: Manipulação de pacotes que criam novas conexões, mudando endereços dos pacotes, origem, destino e etc.
- **mangle**: Regras para alterações nos pacotes

Essas tabelas são utilizadas para administrar regras nas seguintes **chains**:

- **INPUT**: Pacotes com destino ao host local
- **OUTPUT**: Pacotes saindo do host local



- **FORWARD:** Pacotes sendo encaminhados entre redes dentro de um host
- **PREROUTING:** Pacotes antes de se decidir o roteamento
- **POSTROUTING:** Pacotes sendo enviados a uma rede remota após o forward



#### Principais Opções do IPTABLES:

- -A : Adicionar regra na chain
- -I : Inserir regra em uma posição específica da chain
- -R : Substituir regra na chain
- -D : Apagar chain
- -P : Definir política padrão para uma chain
- -L : Listar as regras de uma chain
- -F : Apagar todas as regras de uma chain

#### Criação das Regras no IPTABLES:

- -s endereço (--source) : IP ou Rede Origem do pacote
- -d endereço (--destination) : IP ou Rede Destino do pacote
- -p protocolo (--protocol) : Define o protocolo: tcp, udp, icmp ou all
- -i interface (--in-interface) : Interface de Entrada
- -o interface (--out-interface) : Interface de Saída
- -j ação (--jump) : Ação (Target) para o pacote. Mais comuns ACCEPT, DROP e RETURN

---

### NAT (Network Address Translation)

O iptables pode ser utilizado para criar regras que alteram a origem ou destino dos pacotes, o que é chamado de nat.

A **alteração da origem** é normalmente feita para permitir que uma rede local/privada tenha acesso à Internet.

Nesse caso, as regras podem ser criadas da seguinte maneira:

- `# iptables -t nat -A POSTROUTING -s 172.16.32.0/24 -o eth1 -j SNAT --to-source 200.201.202.203`

- # iptables -t nat -A POSTROUTING -s 172.16.32.0/24 -o eth1 -j **MASQUERADE**

A **alteração do destino** é normalmente feita para realizar o redirecionamento de portas, por exemplo:

- # iptables -t nat -A PREROUTING -p tcp --dport 80 -j DNAT --to-destination 192.168.1.50:8080
- # iptables -t nat -A PREROUTING -p tcp --dport 80 -j REDIRECT --to-port 8080

---

### **Salvando e Restaurando Regras do Iptables**

- iptables-save : Cria uma saída com as atuais regras de firewall, possibilitando que sejam salvos em arquivo.
  - # iptables-save > /etc/iptables.conf
- iptables-restore : A partir de um arquivo, gerado pelo iptables-save, faz um restore das regras.
  - # iptables-restore < /etc/iptables.conf

---

### **IPv6**

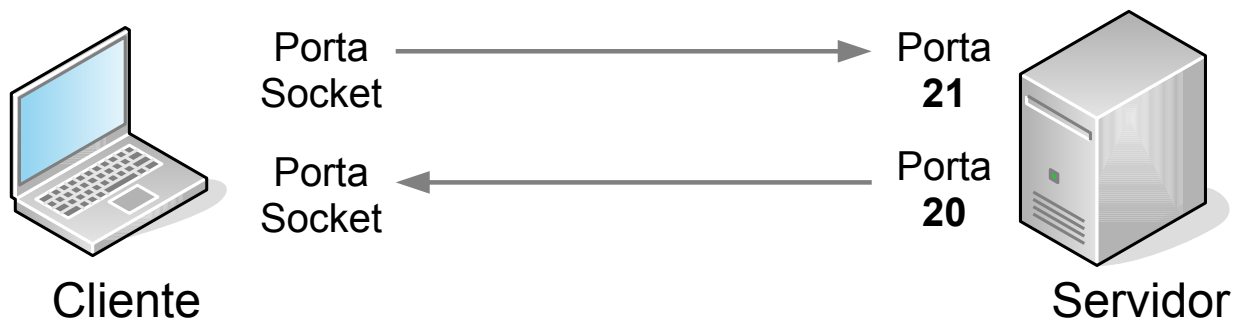
Para o gerenciamento das regras em pacotes IPv6 são utilizados os seguintes comandos:

- ip6tables
- ip6tables-save
- ip6tables-restore

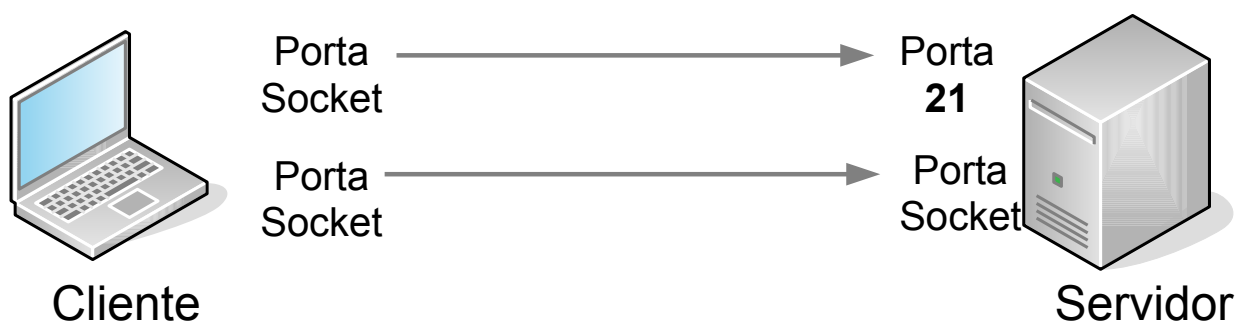
## 212.2 – Protegendo Servidores FTP

### Modos de Conexão Ativo x Passivo

#### Modo Ativo



#### Modo Passivo



---

### vsFTPd (Very Secure FTP)

Arquivo de Configuração: /etc/vsftpd.conf ou /etc/vsftpd/vsftpd.conf

Principais parâmetros de configuração:

- `local_enable` : Habilita/Desabilita o login de usuários locais do sistema
- `anonymous_enable` : Habilita/Desabilita o login de modo anônimo
- `write_enable` : Habilita/Desabilita a alteração de dados via FTP
- `anon_upload_enable`: Habilita/Desabilita o upload de arquivos por usuários anônimos
- `chroot_local_users`: Determina se após o login o usuário ficará ou não preso ao seu diretório

O usuário que executa o processo vsftpd é por padrão o usuário **ftp**. Esse usuário pode ser alterado pelo parâmetro `ftp_username` nos arquivos de configuração.

Para impedir que os usuários visualizem os arquivos presentes no diretório, basta remover a permissão de leitura.

## **Pure-FTPd**

Configurações em:

- /etc/pre-ftp/pure-ftpd.conf (RH/CentOS)
- /etc/default/pure-ftpd-common (Debian)
- /etc/pure-ftpd/conf (Debian)

O processo também pode ser executado via linha de comando.

As principais opções são:

- ChrootEveryone / -A / --chrooteveryone : Determina se após o login o usuário ficará ou não preso ao seu diretório
- AnonymousOnly / -e / --anonymousonly : Determina se o servidor apenas aceitará login de usuários anônimos
- NoAnonymous / -E / --noanonymous : Determina se será bloqueado o login de usuários anônimos
- AnonymousCantUpload / -i / --anonymouscantupload : Determina se será bloqueado o upload de arquivos por usuários anônimos

---

## **ProFTPd (Noções)**

Arquivos de Configuração em /etc/proftdp/proftpd.conf ou /etc/proftpd.conf

## 212.3 – Secure Shell (SSH)

### Configurações do Servidor SSH

As configurações do servidor SSH são implementadas pelo arquivo `/etc/ssh/sshd_config`, as principais são:

- `Port` : Porta utilizada pelo servidor
- `PermitRootLogin` : Define se o login pode ser feito diretamente pelo usuário root
- `PubkeyAuthentication` : Define se o uso de chaves públicas será uma opção de login
- `PasswordAuthentication` : Define se o uso de senhas será uma opção de login
- `ChallengeResponseAuthentication` : Define se será possível o uso do método de múltiplas questões para o login
- `UsePAM` : Define se será possível a autenticação através do uso do PAM
- `PermitEmptyPassword`: Define se serão aceitas senhas em branco
- `Protocol` : Define o protocolo utilizado. Para segurança apenas utilizar o 2.
- `AllowUsers` : Se utilizado, define a lista de usuários que podem fazer login
- `AllowGroupd` : Se utilizado, define a lista de grupos que podem fazer login
- `DenyUsers` : Se utilizado, define a lista de usuários que não podem fazer login
- `DenyGroups` : Se utilizado, define a lista de grupos que não podem fazer login
- `X11Forwarding` : Define se será aceito o acesso a recursos do X11 via SSH
- `AllowTcpForwarding`: Define se será aceito o uso de túneis SSH

---

### Autenticação via Chaves

As chaves públicas dos **servidores conhecidos**, em que já foi feita alguma conexão, são armazenadas localmente no arquivo `~usuario/.ssh/known_hosts` ou `/etc/known_hosts`

Para habilitar o acesso via chave a um servidor deve-se seguir os seguintes passos:

- Criação das chaves público/privada do usuário local:
  - `# ssh-keygen -t <tipo> -b <tamanho>`
    - Serão criados 2 arquivos
      - `~/.ssh/id_rsa` – Chave privada
      - `~/.ssh/id_rsa.pub` – Chave pública
- Enviar a chave pública ao destino:
  - `# ssh-copy-id usuario@IP`
  - A chave ficará armazenada no arquivo `~usuario/.ssh/authorized_keys`

Também pode ser utilizado o login via chaves através do ssh-agent, através dos seguintes passos:

- Criação das chaves público/privada do usuário local:
  - `# ssh-keygen -t <tipo> -b <tamaho>`
  - Definir uma senha para a chave
- Iniciar o ssh-agent
  - `# ssh-agent bash`
- Incluir a chave no ssh-agent
  - `# ssh-add`

## **Túneis Criptográficos com SSH**

Com o SSH é possível criar um túnel criptografado entre 2 hosts de forma que outro serviço não seguro, o utilize para o tráfego de dados. Abaixo a sintaxe:

```
# ssh -N -f -L Porta-Local:IP-Remoto:Porta-Remota usuario@IP-Remoto
```

```
# ssh -N -f -L 4321:192.168.1.210:1234 ricardo@192.168.1.210
```

---

## **SSH e X11**

Acessar via SSH um recurso gráfico via X11:

```
# ssh -X usuario@IP “aplicação”
```

```
# ssh -X ricardo@192.168.1.210 “mousepad”
```

---

## **TCP Wrapper**

Os recursos do TCP Wrapper podem ser utilizados para proteger ainda mais o acesso ao serviço SSH.

Por exemplo:

**/etc/hosts.allow**

```
sshd: 192.168.1.* 10.0.0.10 localhost
```

**/etc/hosts.deny**

```
sshd: ALL
```

## 212.4 – Tarefas de Segurança

### Scan e Testes de Portas

Os seguintes comandos podem ser utilizados para se realizar a varredura e o teste de portas de um host local ou remoto:

- telnet
  - # telnet 192.168.1.1 25
- netcat / nc
  - # nc 192.168.1.1 25
  - # nc -l -p 1234
- netstat
  - # netstat --tcp --listening
  - # netstat -tunl
- nmap
  - # nmap -sT localhost : Conexões TCP abertas
  - # nmap -SU localhost : Conexões UDP abertas
  - # nmap -O : Testes para identificação do Sistema Operacional

\* Revisar o sub-tópico 205.2

---

### Fail2Ban (IPS – Intrusion Prevention System)

O fail2ban monitora arquivos de logs e pode bloquear através do Iptables o acesso de IPs que sejam tentando um acesso indevido a algum serviço.

Principais configurações em **/etc/fail2ban/jail.conf**.

Logs em: /var/log/fail2ban.log

Principais configurações:

- bantime – Tempo de banimento do IP bloqueado
- ignoreip – Lista de IPs que não serão banidos
- maxretry – Quantidade de tentativas de login aceitas antes do banimento
- enabled – Habilita o controle de cada serviço (ssh/telnet/ftp/apache/etc)
- port – Determina o serviço que será monitorado
- logpath – Determina o log que será monitorado para determinado serviço

---

### Snort (NIDS – Network Intrusion Detection System)

Analisa padrões de tráfego que indicam uma possível tentativa de invasão, gerando alertas.

Configurações em /etc/snort/snort.conf

Regras/Assinaturas em /etc/snort/rules

## **OpenVAS (Open Vulnerability Assessment System)**

- Voltado para Pentests (Testes de Invasão)
- Trabalha no Modelo Cliente Servidor
- Servidor armazena um conjunto de testes de vulnerabilidade e os executa regularmente nos clientes
- NVT = Network Vulnerability Tests

---

## **Boletins de Segurança**

As principais fontes de informações de incidentes de segurança são:

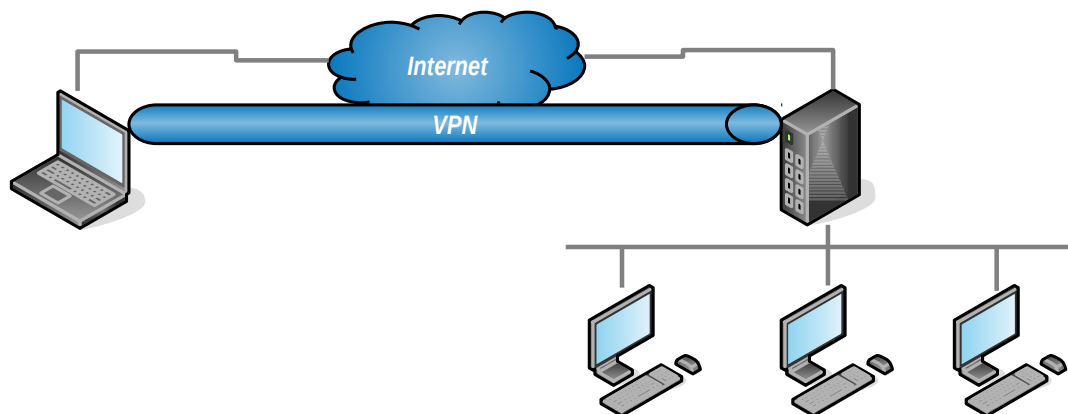
- **Bugtraq:** [www.securityfocus.com](http://www.securityfocus.com) / <https://www.securityfocus.com/archive/1>
- **CERT:** <https://www.us-cert.gov/>, [www.cert.br](http://www.cert.br)



## 212.5 – OpenVPN

### VPN – Virtual Private Network

- Túnel Criptografado entre dois ou mais elementos de rede, normalmente conectados por uma rede pública.
- Em ambientes GNU/Linux, a aplicação mais popular é o OpenVPN



---

### OpenVPN

- Utiliza port UDP 1194
- Tipos de Encriptação:
  - Static Key – Cliente e Servidor usam a mesma chave
  - Public Key – São gerados certificados e chaves que são compartilhadas
- Tanto na origem quanto no destino haverá uma interface virtual para a criptografia dos dados, referenciadas como tun ou tap.
  - tap – Nível ethernet (layer 2). Usado para Bridges.
  - tun – Nível de rede (layer 3). Roteamento IP.

A chave estática pode ser gerada pelo próprio openvpn pelo comando:  
`# openvpn --genkey --secret static.key`

Configurações presentes em `/etc/openvpn`. Principais parâmetros:

- `dev` : Define a interface como tap ou tun
- `ifconfig` : Define as configurações de IP da rede criptografada
- `secret` : Define a localização do arquivo que contém a chave estática utilizada na conexão
- `push` (apenas servidor) : Envia comandos DHCP após uma conexão de cliente
- `status` : Define um arquivo em que quando usada no cliente ou em instâncias rodando no modo point-to-point, o arquivo gerado conterá estatísticas de tráfego. Já em instâncias funcionando como servidores conterá a lista de clientes conectados e a tabela de roteamento.
- `remote` (apenas cliente) : IP do Servidor VPN em que o cliente se conectará
- `nobind` (apenas cliente) : Alocar uma porta dinâmica ao invés da 1194 na conexão do cliente
- `client-to-client`: Faz com que seja criado um roteamento direto entre os clientes VPN, ao invés de enviar todo o tráfego de origem do cliente para a interface tun/tap

O processo pode ser iniciado das seguintes maneiras:

- `# openvpn server.conf &`
- `# openvpn --config client.conf --daemon`