DlteC do Brasil®
www.dltec.com.br
info@dltec.com.br|413045.7810

DLTEC DO BRASIL

CURSO SERVIDOR DE NOMES DE DOMÍNIO NO LINUX (TÓPICO 207 DA LPI 202)

Curso Servidor de Nomes de Domínio No Linux

O conteúdo dessa apostila é uma adaptação da matéria online do Curso Servidor de Nomes de Domínio no Linux.

Para mais informação visite www.dltec.com.br

Introdução

Olá!

Como parte integrante do curso **Linux LPI-202** da Dltec do Brasil, esta apostila representa uma adaptação textual do material disponibilizado online pelo curso. Por isso, recomendamos que você utilize-a como um importante recurso offline. Combinando-a com o conteúdo online, você estará muito melhor preparado(a) para realizar o exame 202-450 do LPI.

É de suma importância que você, além de participar dos fóruns, realize o máximo possível de exercícios e simulados. Para iniciar, assista aos vídeos introdutórios do capítulo 01 (disponibilizados na Área do Aluno). Lá você obterá mais detalhes sobre o funcionamento geral do curso.

Esperamos que você aproveite ao máximo este material, que foi idealizado com o intuito verdadeiro de fazê-lo(a) obter êxito no exame. Estamos torcendo pelo seu sucesso!

Bons estudos!

Neste primeiro curso da trilha LPI 202, estudaremos o tópico **207 – Servidor de Nomes de Domínio no Linux**, composto por três subtópicos importantes.

Como oito das 60 questões do exame sairão de assuntos aqui abordados (cerca de 13%), é necessário que você esteja muito atento(a) e revise-o constantemente.

Pratique os exercícios propostos no curso e crie os seus resumos. Utilize o Oracle VirtualBox para criar máquinas virtuais e treinar os comandos apresentados.

A experiência "hands-on" exercerá um impacto superpositivo no dia do seu exame. Portanto, pratique constantemente o que você aprenderá por aqui.

É também importante que você sempre tenha uma coisa em mente: como está subindo na hierarquia do LPI, o nível de cobrança das questões também é superior. Portanto, não tenha pressa no processo de aprendizagem. O mais importante é: somente passe para o próximo assunto quando o anterior estiver compreendido – no nível requisitado no exame.

A Dltec está contido nessa jornada!

Bons estudos!

Servidor de Nomes de Domínio no Linux

Peso: 8

Objetivos do Curso

Ao final desse curso você deverá ter conhecimentos sobre:

- Configuração básica de um servidor de DNS.
- Criar e manter zonas de DNS.
- Garantir a segurança de um servidor de DNS.

Sumário

1 Introdução

1.1 Introdução ao Curso

Bem-vindo ao curso sobre **Servidor de Nomes de Domínio no Linux**, o qual também faz parte do conteúdo preparatório para a prova de certificação **LPIC-2**, mais especificamente do exame LPI-202, cujo código é **202-450**.

Ao final desse curso você deverá ter conhecimentos sobre:

- Configuração básica de um servidor de DNS.
- Criar e manter zonas de DNS.
- Garantir a segurança em um servidor de DNS.

Mesmo que não esteja trilhando os estudos para a certificação, você poderá fazer este curso para aumentar os seus conhecimentos no mundo Linux.

Mas se você está na trilha da certificação, saiba que esse curso aborda o **Tópico 207** da prova que corresponde a cerca de **13% das questões do exame 202-450** e tem **peso 8** nesta certificação.

A seguir, vamos falar mais sobre as provas da LPI, da certificação **LPIC-2** e como tudo isso está planejado aqui em nosso Portal.

Não esqueça de que, ao final do curso, você poderá emitir o seu certificado!

1.2 Sobre a LPI e a LPIC-2

"A maior e mais reconhecida certificação Linux do mundo".

A **Linux Professional Institute (LPI)** é a organização mais respeitada no credenciamento de profissionais Linux do mundo, sendo adotada por várias empresas como o padrão global de certificação e organização de suporte de carreira para **profissionais do mundo Linux.**

Atualmente eles possuem **mais de 175,000 profissionais certificados** em **mais de 180 países**.

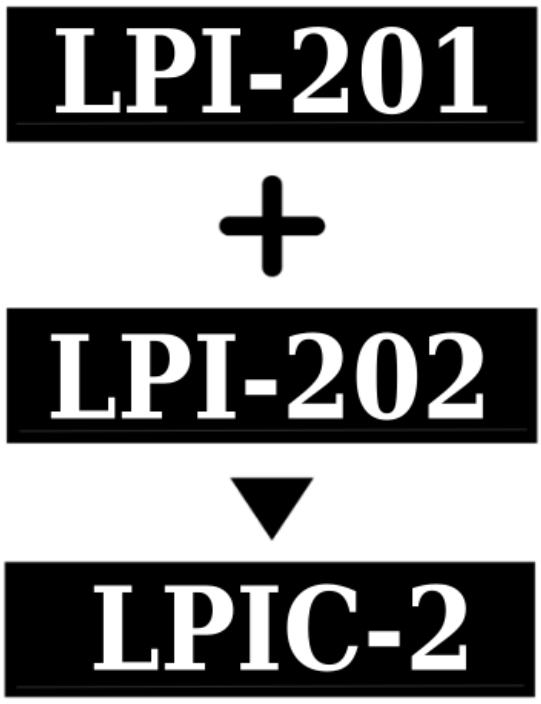
Suas certificações são denominadas de "LPIC" ou "Linux Professional Institute Certification" e, mais especificamente, a certificação **LPIC-2** é a segunda a ser realizada no programa de certificação profissional Linux multi-nível da **LPI ou Linux Professional Institute**.

Para seu conhecimento, a **LPI** prevê atualmente TRÊS níveis de certificação, ou seja, LPIC-1, **LPIC-2** e LPIC-3.

Após passar por um rigoroso processo de desenvolvimento, a certificação **LPIC-2** traz em seus objetivos diversas competências a serem testadas nos candidados que desejam trabalhar com o Linux em um nível muito mais estratégico – mais avançado do que a anterior.

Por esta razão, a **LPIC-2** validará a sua capacidade como candidato em administrar redes de comunicação de pequeno a médio porte, incluindo cenários onde o Linux seja utilizado em conjunto com outros sistemas – compondo, assim, um ambiente misto.

A versão atual da certificação **LPIC-2** é a **4.5**, composta por duas provas com códigos de exame **"201-450"** (ou **"LPI-201"**) e **"202-450"** (ou **"LPI-202"**).



O único requisito para obter a certificação **LPIC-2** é que você possua a certificação **LPIC-1** ainda ativa. Porém, os exames necessários para obtê-las poderão ser realizados em qualquer ordem. Em outras palavras: mesmo que você ainda não seja certificado **LPIC-1**, você poderá fazer ambos os exames da **LPIC-2**. No entanto, você somente receberá esta certificação quando for aprovado(a) nos exames requeridos pela **LPIC-1**.

A certificação **LPIC-2** tem **validade de 5 anos** e um dos seus diferenciais é o fato dela poder ser **realizada em Português**!

Atualmente os exames da **LPIC-2** podem ser realizada nos idiomas Inglês, Alemão, Japonês e **Português** (Brasileiro).

Os exames da LPI podem ser realizados em centros autorizados da **Pearson Vue**, sendo necessário cadastro e matrícula para a realização do exame diretamente pelo site da **Pearson Vue**. O processo de cadastramento e aquisição dos exames serão demonstrados dentro do preparatório durante o curso.

Para tornar-se certificado **LPIC-2** você deverá ser capaz de:

- Realizar tarefas administrativas avançadas, incluindo aquelas relacionadas ao kernel Linux, inicialização e manuntenção geral do sistema.
- Efetuar o gerenciamento avançado de dispositivos de bloco e de sistemas de arquivos, além de solucionar problemas de rede envolvendo a autenticação de usuários e questões sobre segurança (incluindo firewall e VPN).
- Instalar e configurar serviços de rede fundamentais, como DHCP, DNS, SSH, servidores Web, servidores de arquivos usando FTP, NFS e Samba – além de serviços de email.
- Supervisionar assistentes e aconselhar os níveis gerenciais sobre medidas a serem tomadas relativas a automação e compras.

A seguir vamos falar sobre como a preparação para a **LPIC-2** está dividida no **Portal da DlteC** e como você deverá utilizar nossa material para conquistar sua certificação.

1.3 Plano de Estudos para a LPIC-2 e LPI-202

Como você já sabe, a certificação **LPIC-2** é composta por DOIS exames: **LPI-201 (prova 201-450)** e **LPI-202 (prova 202-450)**.

Este curso em que você está matriculado faz parte da trilha de preparação da **LPI-202**.

Existe um curso no Portal chamado **"LPIC-2: LPI 202-450"**, o **qual agrega todos os cursos da trilha da LPI-202** e que contém os simulados e fóruns de tira dúvidas com o professor responsável pelo curso.

O **plano de estudos** para você ter sucesso na **LPI-202** é o seguinte:

1. Estudar o conteúdo de cada Curso Express (sequência de cursos a seguir).
2. Repetir os comandos e demonstrações práticas realizadas pelo Prof. Leandro durante as vídeo aulas como laboratório.
3. Fazer os simulados que estão dentro do curso **"LPIC-2: LPI 202-450"**.
4. A qualquer momento tirar as dúvidas do conteúdo utilizando os fóruns correspondentes.

A **LPI-202** é composta dos cursos conforme sequência da figura abaixo:

O exame **LPI-202** é composto por 60 questões, a serem resolvidas em 90 minutos.

Cada um dos tópicos que compõem este exame possui os seus **Objetivos**; cada objetivo possui um **Peso**.

Quanto maior o peso, maior será a quantidade de questões no exame.

Para ser aprovado(a), você deverá conseguir obter (no mínimo) 500 pontos na **LPI-202**.

Se você fez a matrícula nesse curso com o objetivo de tirar a certificação então a partir de agora, foco total no objetivo: **OBTER A CERTIFICAÇÃO.**

Você será aprovado(a) – já coloque isso "na cabeça".

Para isso, pratique os comandos, leia os tópicos com cautela e marque logo o dia do seu exame (para você já ter uma data limite).

Faça o seu cronograma, estipule as horas de estudo e, sinceramente, não tem erro.

Repita essa frase todos os dias: **Eu serei aprovado(a).**

Se você assumir esse compromisso com sinceridade e vontade de vencer, **tudo dará certo**.

Estamos ao seu lado! Bons estudos!

(*) Os fóruns do curso são exclusivos para TIRAR AS DÚVIDAS DO CURSO, caso você tenha dúvidas do dia a dia ou que não tenham correlação com o curso utilize os grupos do Facebook ou Telegram para troca de ideias.

1.4 Como Estudar

Nesse curso você terá **vídeo-aulas** e **material de leitura** para o aprendizado do conteúdo.

Posso somente ler ou assistir aos vídeos? NÃO RECOMENDAMOS!

O ideal é você assistir aos vídeos e na sequência ler os conteúdos ou...

... se preferir leia os conteúdos e depois assistia aos vídeos, tanto faz.

POR QUE LER E ASSISTIR?

Simples, porque **um conteúdo complementa o outro**. Principalmente se você está se preparando para a prova de certificação é crucial que você tanto veja os vídeos como a matéria de leitura!

2 Configuração Básica de um Servidor de DNS

2.1 Introdução

Subtópico: 207.1 Peso: 3

Descrição do Objetivo: Você deverá saber configurar o BIND para atuar como um servidor de DNS "caching-only". Este objetivo inclui a habilidade de gerenciar um servidor em execução e a configuração de registros de log.

Áreas-Chave:

- Arquivos de configuração, termos e utilitários do BIND 9.x.
- Definição da localização dos arquivos de zona do BIND em seus arquivos de configuração.
- Recarregamento de arquivos de zona e de configuração modificados.
- Ciência a respeito do dnsmasq, djbdns e do PowerDNS como alternativas de servidores de nomes.

Principais Termos, Arquivos e Utilitários:

- /etc/named.conf
- /var/named
- /usr/sbin/rndc
- kill
- host
- dig

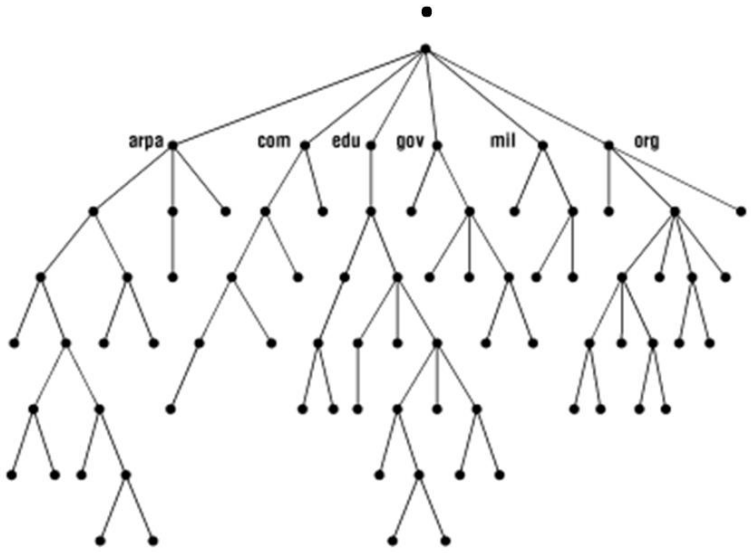
2.2 Conceitos Introdutórios e Comandos Básicos

O **DNS** (Domain Name System) é um sistema constituído por uma robusta, hierárquica e distribuída base de dados, cujo propósito é criar mapeamentos entre nomes de hosts com endereços IP (e vice versa).

Esta base utilizada pelo sistema é indexada a partir de nomes de domínios, representados por caminhos lógicos baseados em uma árvore invertida, conhecida como Domain Name Space.

No topo desta árvore encontramos uma única raiz (denominada **root domain**), gerenciada pela **ICANN** (Internet Corporation for Assigned Names and Numbers) e representada pelo ponto (**.**).

Com isto, os nomes de domínios são sempre lidos a partir desta raiz. Observe na figura a seguir uma representação parcial do Domain Name Space:



No **root domain** são encontrados diversos **root servers**, localizados em diferentes localizações do globo e classificados em 13 grandes autoridades.

Aliás é importante associar a idéia de "domínio" a um nome (facilmente assimilado por humanos) que possui como objetivo básico ser utilizado para disponibilizar algum recurso pela grande rede. Estes também podem ser categorizados, variando de acordo com o seu nível.

Os Top-Level Domains são categorizados de formas diferentes pela **IANA** (Internet Assigned Numbers Authority). Por exemplo, alguns daqueles considerados genéricos (gTLD) são descritos a seguir:

- **com**: utilizado principalmente por organizações comerciais.
- **edu**: utilizado por instituições de ensino superior.
- **org**: originalmente era utilizado por organizações não comerciais, mas, ainda durante a década de 1990, essa restrição foi removida.
- **net**: originalmente era utilizado por organizações relacionadas a infraestrutura de redes, mas, ainda durante a década de 1990, também encontra-se disponível para ser utilizado por organizações comerciais.
- **mil**: utilizado por organizações militares.
- **gov**: Uso governamental.

A **IANA** também classifica os TLDs segundo os códigos utilizados por países – esses são conhecidos como Country-Code TLD (ou ccTLD). Os caracteres utilizados para identificar os países são baseados no padrão ISO 3166. Daí entram em cena o **br** (Brasil), **fr** (França), **de** (Alemanha), **uk** (Reino Unido), **us** (Estados Unidos) dentre outros.

A gestão do **br**, por exemplo, é realizada pela associação **NIC.br** (Núcleo de Informação e Coordenação do PontoBR), criada por membros do **CGI.br** (Comitê Gestor da Internet no Brasil).

Curiosidade:

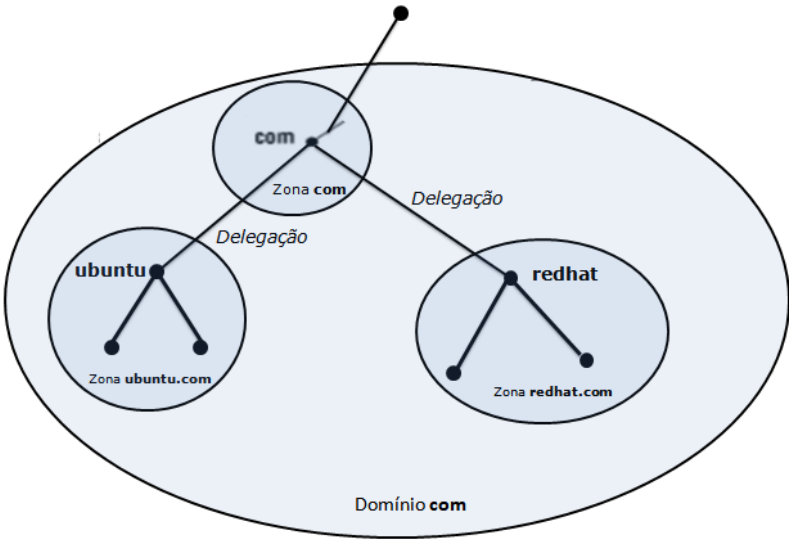
No início da Internet, os mapeamentos entre nomes e endereços IP eram mantidos pelo Network Information Center (**NIC**) em apenas um único arquivo – conhecido como HOST.TXT. Este era distribuído através do protocolo **FTP** aos outros hosts conectados à grande rede mundial.

Os subdomínios diretos dos TLDs são conhecidos como First Level Domains (FLDs), destinados a organizações, indivíduos etc. Por exemplo, o TLD "com" possui subdomínios como "google.**com**", "globo.**com**", "redhat.**com**", "linuxmint.**com**" dentre outros.

Através do processo conhecido como **delegação de autoridade**, a gestão do FLDs poderá ser realizada pelos seus próprios mantenedores. Dito isto, a responsabilidade pela gestão do domínio "standford.edu" é delegada à própria Standford University, por exemplo.

Dentro de um FLD, o seu mantenedor poderá definir hosts a serem localizados de forma própria. Por exemplo, geralmente os servidores web são acessíveis através da definição do "host" www – ou utilizar, por exemplo, um "host" ftp para disponibilizar este serviço no domínio (como ftp.exemplo.com) etc.

O sistema de **DNS** pode ser subdividido em zonas diferentes. Essas zonas são partes do Domain Name Space que são gerenciadas por organizações específicas. Além disso, cada zona poderá possuir muitos domínios – da mesma forma que, no mesmo servidor de **DNS**, várias zonas poderão co-existir.



Os servidores de nomes (**nameservers**) atuantes em uma zona, conhecem todas as informações a respeito desta, já que esses detalhes são obtidos a partir de arquivos locais ou vindos a partir de outro **nameserver**. Nesse contexto, dizemos que tratam-se de **nameservers** autoritativos (Authoritatives).

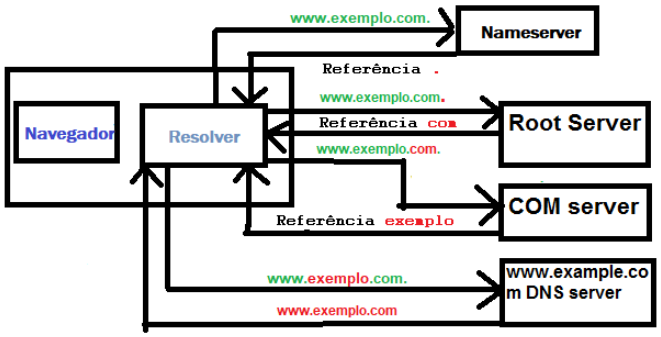
Veja na figura anterior que o domínio **com** possui, em seu topo, a zona de mesmo nome. Dessa forma, a gestão do subdomínio "redhat.com", por exemplo, não é de responsabilidade do **com**, mas sim da própria equipe do Red Hat. Porém, os **nameservers** atuantes no domínio **com** sabem como obter informações do "redhat.com", já que trata-se de um dos seus subdomínios.

Os nameservers autoritativos também poderão assumir comportamentos diferentes em uma zona. Aquele que é utilizado para armazenar os arquivos originais a respeito da zona, é identificado como **Master** (ou Primary). No processo de definição da zona neste **nameserver**, a zona também é especificada como "master". Devido a possuir os registros "master" da zona, este deverá sempre estar presente.

Já quando um **nameserver** obtém as informações sobre uma zona a partir do **Master**, dizemos que trata-se de um **nameserver Slave**. Dessa forma, esse mantém uma cópia idêntica dos dados contidos no primeiro. O processo de transferência dessas informações é conhecido Transferência entre Zonas.

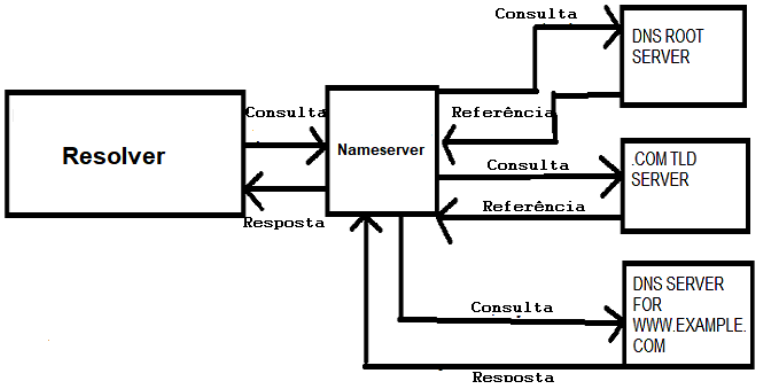
No decorrer do curso conheceremos algumas variações que poderão ser assumidas por um **nameserver**. Aliás, esses também se diferenciam pela forma em que manipulam as requisições de consultas que lhes são enviadas pelos resolvedores (**resolvers**) do sistema operacional.

Ao ser configurado para manipular **consultas iterativas** (ou não-recursivas) o **nameserver** não se responsabiliza por oferecer uma resposta completa à consulta do cliente – ele devolve uma referência de outros servidores de **DNS** que poderão ser usados na busca aplicada pelo **resolver** (caso o **nameserver** não possua a resposta para a pergunta, obviamente).



Um **resolver** é um programa ou rotina responsável por formular uma consulta de **DNS** (também conhecido como cliente de **DNS**).

Quando configurado para atender **consultas recursivas** o processo é inverso: o **nameserver** irá realizar todos os procedimentos até conseguir atender à requisição que lhe fôra passada. Neste cenário, ele poderá até mesmo consultar os outros servidores:



Alguns comandos oferecidos pelo Linux são bastante úteis ao trabalharmos com **DNS**. O primeiro a ser destacado é o **nslookup**. Apesar de ser considerado obsoleto, ainda costuma ser utilizado para efetuar consultas aos **nameservers**.

Por exemplo, se desejamos obter detalhes a respeito do domínio "centos.org", lançamos o **nslookup** da seguinte forma:

```
[root@curso8 ~]#  nslookup centos.org

Server:         192.168.0.1
Address:        192.168.0.1#53

Non-authoritative answer:
Name:           centos.org
Address:        81.171.33.201
Name:           centos.org
Address:        81.171.33.202
Name:           centos.org
Address:        2001:4de0:aaae::201
Name:           centos.org
Address:        2001:4de0:aaae::202
```

Note que o campo Server indica o IP do servidor que foi utilizado para manipular o procedimento de consulta. Note também que os resultados exibidos são antecedidos pela string Non-authoritative answer. Isto significa que o **nameserver** local não possui qualquer autoridade sobre o nome de domínio que lhe fora passado – as informações exibidas foram obtidas a partir de **nameservers** externos.

Desta forma, podemos por exemplo informar agora ao comando **nslookup** para que use um servidor de **DNS** específico (ao contrário daquele disponível em **/etc/resolv.conf**):

```
[root@curso8 ~]#  nslookup centos.org 8.8.4.4

Server:         8.8.4.4
Address:        8.8.4.4#53

Non-authoritative answer:
Name:   centos.org
Address: 81.171.33.201
Name:   centos.org
Address: 81.171.33.202
Name:   centos.org
Address: 2001:4de0:aaae::201
Name:   centos.org
Address: 2001:4de0:aaae::202
```

Outro comando simples porém bastante objetivo é o **host**:

```
[root@curso8 ~]#  host centos.org

centos.org has address 81.171.33.201
centos.org has address 81.171.33.202
centos.org has IPv6 address 2001:4de0:aaae::202
centos.org has IPv6 address 2001:4de0:aaae::201
centos.org mail is handled by 10 mail.centos.org.
```

Note que, ao contrário do **nslookup**, por padrão o comando também exibe informações a respeito do servidor responsável por manipular e-mails no domínio especificado. Da mesma forma que o anterior, também podemos especificar um determinado servidor de **DNS** a ser consultado:

```
[root@curso8 ~]#  host centos.org 8.8.8.8

Using domain server:
Name: 8.8.8.8
Address: 8.8.8.8#53
Aliases:

centos.org has address 81.171.33.201
centos.org has address 81.171.33.202
centos.org has IPv6 address 2001:4de0:aaae::201
centos.org has IPv6 address 2001:4de0:aaae::202
centos.org mail is handled by 10 mail.centos.org.
```

Já o comando **dig** (Domain Information Groper) é aquele que apresenta as mais diversas possibilidades de uso. Bastante flexível e robusto, o comando é um dos mais utilizados em procedimentos de troubleshooting relacionados a **DNS**:

```
[root@curso8 ~]#  dig centos.org @8.4.4.4

; <<>> DiG 9.11.4-P2-RedHat-9.11.4-9.P2.el7 <<>> centos.org @8.8.4.4
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 30874
;; flags: qr rd ra; QUERY: 1, ANSWER: 2, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 512
;; QUESTION SECTION:
;centos.org.                    IN      A

;; ANSWER SECTION:
centos.org.             576     IN      A       81.171.33.202
centos.org.             576     IN      A       81.171.33.201

;; Query time: 32 msec
;; SERVER: 8.8.4.4#53(8.8.4.4)
;; WHEN: Fri Jan 17 18:50:43 -03 2020
;; MSG SIZE  rcvd: 71
```

Note que o comando exibe no campo Query time o tempo total que a consulta demorou para ser concluída, além de também informar o servidor que foi utilizado no processo.

Veja também que, através da seção Authority Section, o comando informa os nomes dos servidores que foram utilizados para oferecer respostas autoritativas a respeito do domínio que foi informado. Na seção seguinte, são exibidos os seus endereços IP.

Iremos explorar bastante esses três comandos no decorrer do curso. A sugestão é que você abra agora mesmo as páginas de seus manuais e já inicie a leitura, já que isso eleverá bastante o seu desempenho para o exame.

O **BIND** (Berkeley Internet Name Domain) é a solução open source mais utilizada para implementar o serviço de **DNS**. Devido a sua importância, o **LPI** espera que você saiba manipulá-lo devidamente, atentando-se a conceitos-chave e aspectos configurativos. Nosso foco se recairá sobre a versão **9.x**.

Entretanto, existem outras soluções que agem como alternativas ao **BIND** e que você precisa ter ciência no dia exame.

A primeira que poderá ser destacada é o **djbdns**, escrita por Daniel J. Bernstein para ir de encontro às brechas de segurança que ele encontrava no **BIND**. Na verdade, o **djbdns** é uma suíte composta por diferentes soluções, que inclui o tinydns e o dnscache. O projeto foi abandonado no início da década de 2000. Apesar disso, outros programas surgiram como um fork do djbdns, como é o caso do dbndns, do Debian.

O **dnsmasq** é outra solução bastante leve e de fácil configuração, utilizada como um **nameserver Forwarder**, ou seja, sua finalidade é encaminhar consultas de **DNS** a outros servidores de nome. Dito de outra forma, eles "terceirizam" o processo de resolução de nomes. O dnsmasq também consegue atuar como um servidor de **DHCP**. Por essas características, esta solução costuma ser bastante utilizada em roteadores.

O **PowerDNS** é outra solução modular open source bastante utilizada, lançada no início da década de 2000 sob a licença GPL. Apesar de original ser lançado como um software proprietário, posteriormente seus códigos se tornaram públicos. Além de trabalhar com os tradicionais arquivos de zona, esta solução também consegue interagir com sistemas de gerenciamento de bancos de dados complexos, como o Oracle, MySQL e MariaDB.

Como o nosso foco será o **BIND**, vamos utilizar um host CentOS para a sua instalação. Este host será identificado nos exemplos tanto pelo seu IP (192.168.0.15) quanto pelo seu hostname (LAP1). Sendo assim, vamos instalar o pacote "bind":

```
[root@curso8 ~]#  yum install bind
```

O principal arquivo de configuração do **BIND** é identificado como **named.conf**, contendo as instruções globais aplicadas ao serviço.

No CentOS, ele encontra-se posicionado diretamente sob o diretório **/etc**. Como veremos mais a frente, no Debian este mesmo arquivo encontra-se sob **/etc/bind**. Atente-se para essa questão no dia do exame.

Por padrão, este arquivo é disponibilizado com diversos comentários – seja de apenas uma linha (iniciando com os caracteres "//" ou de um conjunto de linhas (iniciando com os caracteres "/*" e finalizando com "*/". Vamos visualizar o seu conteúdo vindo por padrão:

```
options {
listen-on port 53 { 127.0.0.1; };
      listen-on-v6 port 53 { ::1; };
      directory       "/var/named";
      dump-file       "/var/named/data/cache_dump.db";
      statistics-file "/var/named/data/named_stats.txt";
      memstatistics-file "/var/named/data/named_mem_stats.txt";
      recursing-file  "/var/named/data/named.recursing";
secroots-file   "/var/named/data/named.secroots";
      allow-query     { localhost; };
recursion yes;
      dnssec-enable yes;
      dnssec-validation yes;
      bindkeys-file "/etc/named.root.key";
      managed-keys-directory "/var/named/dynamic";
      pid-file "/run/named/named.pid";
      session-keyfile "/run/named/session.key";
};

logging {
            channel default_debug {
            file "data/named.run";
            severity dynamic;
};
};

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

include "/etc/named.rfc1912.zones";
include "/etc/named.root.key";
```

Ao manipularmos os **nameservers** no decorrer do curso, iremos estudar muitas das instruções que estão contidas neste arquivo.

2.3 Implementação do Servidor de DNS Caching-Only

Nosso primeiro objetivo será aprender a implementar um **nameserver** para atuar como **Caching-Only**. Desta forma, após as consultas serem resolvidas, este irá armazenar as respostas em cache para acelerar consultas similares futuras.

Sendo assim, este **nameserver** não irá atender outras zonas de **DNS** (exceto aqueles nele posicionadas). Sua função é rigorosamente esta: realizar consultas e mantê-las em cache.

Ao ser instalado, o **BIND** já vem programado para atuar como um servidor **caching-only**. Porém, precisamos entender o quê caracteriza este tipo de atuação em relação a critérios de configuração. Vamos inicialmente destacar alguns pontos relativos ao bloco de instrução **options**:

```
listen-on port 53 { 127.0.0.1; };

directory       "/var/named";

allow-query     { localhost; };

recursion yes;
```

A instrução **listen-on** determina a porta em que este serviço estará em atuação. Como vemos, o serviço irá atender à porta 53. Entre as chaves, podemos encontrar IPs que estarão ouvindo na porta em questão. Neste momento, apenas o localhost está configurado para ouvi-la.

Já **directory** é responsável por especificar o diretório de trabalho do **BIND**. Neste caso, por padrão, o serviço utiliza o **/var/named**.

A instrução **Allow-query** é responsável por controlar quais hosts poderão efetuar consultas a este **nameserver**. Por enquanto, somente serão atendidas consultas efetuadas pelo localhost.

Por fim, a instrução **recursion** permite indicar se o **nameserver** poderá ou não manipular consultas recursivas.

Como este **nameserver** deverá atuar neste momento como **Caching-Only**, a recursividade é algo imprescindível, já que precisará consultar diferentes **nameservers** para oferecer respostas relacionadas a domínios por ele desconhecidos.

Nesse cenário, o **nameserver** precisará saber como acessar os **root servers**. Isto é configurado através da única instrução **zone** vinda por padrão neste arquivo:

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

Note que o tipo desta zona é identificado como **hint**. Este tipo indica que trata-se de uma zona onde diferentes **root servers** poderão ser acessados. Para o exame, memorize a sintaxe referente a esta zona.

Esses **root servers** podem ser visualizados através do arquivo **named.ca**. Como não possui um caminho especificado na instrução **file**, utiliza-se o diretório configurado na instrução **directory**. Sendo assim, este arquivo localiza-se sob **/var/named**. Observe o seu conteúdo parcial:

```
; <<>> DiG 9.11.3-RedHat-9.11.3-3.fc27 <<>> +bufsize=1200 +norec @a.root-
servers.net
; (2 servers found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 46900
;; flags: qr aa; QUERY: 1, ANSWER: 13, AUTHORITY: 0, ADDITIONAL: 27

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 1472
;; QUESTION SECTION:
;.                              IN      NS

;; ANSWER SECTION:
.                       518400  IN      NS      a.root-servers.net.
.                       518400  IN      NS      b.root-servers.net.
.                       518400  IN      NS      c.root-servers.net.
.                       518400  IN      NS      d.root-servers.net.
.                       518400  IN      NS      e.root-servers.net.
.                       518400  IN      NS      f.root-servers.net.
.                       518400  IN      NS      g.root-servers.net.
.                       518400  IN      NS      h.root-servers.net.
.                       518400  IN      NS      i.root-servers.net.
.                       518400  IN      NS      j.root-servers.net.
.                       518400  IN      NS      k.root-servers.net.
.                       518400  IN      NS      l.root-servers.net.
.                       518400  IN      NS      m.root-servers.net.

;; ADDITIONAL SECTION:
a.root-servers.net.     518400  IN      A       198.41.0.4
b.root-servers.net.     518400  IN      A       199.9.14.201
```

```
c.root-servers.net.     518400  IN      A       192.33.4.12
d.root-servers.net.     518400  IN      A       199.7.91.13
e.root-servers.net.     518400  IN      A       192.203.230.10
f.root-servers.net.     518400  IN      A       192.5.5.241
g.root-servers.net.     518400  IN      A       192.112.36.4
h.root-servers.net.     518400  IN      A       198.97.190.53
i.root-servers.net.     518400  IN      A       192.36.148.17
j.root-servers.net.     518400  IN      A       192.58.128.30
k.root-servers.net.     518400  IN      A       193.0.14.129
```

Note que o seu conteúdo é resultante da execução do comando **dig**.

Vamos agora utilizar alguns comandos para visualizarmos algumas questões importantes na prática.

Estando no próprio LAP1, vamos editar o seu arquivo **/etc/resolv.conf** para determinar que a resolução de nomes seja feita pelo localhost:

```
[root@curso8 ~]#  echo "nameserver 127.0.0.1" > /etc/resolv.conf
```

Para que consigamos trabalhar de forma efetiva, precisamos agora parar e desabilitar o daemon **firewalld**, caso este esteja disponível, obviamente:

```
[root@curso8 ~]#  systemctl stop firewalld && systemctl disable firewalld
```

Em seguida, vamos iniciar o serviço do **BIND**. Em distribuições baseadas no Red Hat, o daemon associado ao serviço é conhecido como **named**. Sendo assim, aplicamos o comando **systemctl**:

```
[root@curso8 ~]#  systemctl start named
```

Neste momento podemos lançar o comando **rndc** (Remote Name Daemon Control) para verificar o estado geral do serviço. Como ele é responsável por controlar diferentes aspectos do daemon **named**, diferentes subcomandos poderão ser aplicados. Neste momento, utilizaremos o **status**:

```
[root@curso8 ~]#  rndc status

version: BIND 9.11.4-P2-RedHat-9.11.4-9.P2.el7 (Extended Support Version)
<id:7107deb>
running on lap1: Linux x86_64 5.4.6-Dltec26122019 #1 SMP Thu Dec 26 18:30:53 -03
2019
boot time: Thu, 16 Jan 2020 19:09:41 GMT
last configured: Thu, 16 Jan 2020 19:09:41 GMT
configuration file: /etc/named.conf
CPUs found: 1
worker threads: 1
UDP listeners per interface: 1
number of zones: 103 (97 automatic)
debug level: 0
xfers running: 0
xfers deferred: 0
soa queries in progress: 0
query logging is OFF
recursive clients: 0/900/1000
tcp clients: 2/150
server is up and running
```

Note que, além de exibir a versão do **BIND** que encontra-se instalada, é também exibido que o daemon já encontra-se ativo e em execução.

Sendo assim, vamos utilizar o comando **dig** para visualizarmos informações sobre o domínio redhat.com:

```
[root@curso8 ~]#  dig redhat.com

; <<>> DiG 9.11.4-P2-RedHat-9.11.4-9.P2.el7 <<>> redhat.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 36475
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 6, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;redhat.com.                    IN      A

;; ANSWER SECTION:
redhat.com.             3600    IN      A       209.132.183.105

;; AUTHORITY SECTION:
redhat.com.             172793  IN      NS      a9-65.akam.net.
redhat.com.             172793  IN      NS      a13-66.akam.net.
redhat.com.             172793  IN      NS      a1-68.akam.net.
redhat.com.             172793  IN      NS      a28-64.akam.net.
redhat.com.             172793  IN      NS      a10-65.akam.net.
redhat.com.             172793  IN      NS      a16-67.akam.net.

;; Query time: 3376 msec
;; SERVER: 127.0.0.1#53(127.0.0.1)
;; WHEN: Thu Jan 16 16:11:31 -03 2020
;; MSG SIZE  rcvd: 187
```

Note que o **nameserver** conseguiu resolver esta consulta em 3376 milisegundos. Se lançarmos o mesmo comando novamente, a consulta irá ser realizada de forma muito mais veloz, já que esta resposta já encontra-se em sua área de cache.

Sendo assim, este é o comportamento do **nameserver Caching-Only**. Memorize essas configurações para o dia do exame.

Vamos realizar alguns testes para entendermos ainda mais sobre as configurações do **BIND**. Para isto, vamos editar o arquivo **/etc/named.conf** e alterar as seguintes instruções contidas no bloco **options**:

```
listen-on port 53 { 127.0.0.1; 192.168.0.15; };

allow-query { any; };
```

Veja que, além do 127.0.0.1, agora também incluímos o endereço IP do host na instrução **listen-on** para que este também esteja ouvindo à porta 53. Além disso, alteramos a instrução **allow-query** para permitir que qualquer outro host consiga efetuar consultas a este **nameserver**.

Após salvar e sair da edição do arquivo, podemos lançar o comando **named-checkconf** sobre ele, de forma que sejam analisadas as configurações lá presentes e aponte eventuais erros de sintaxe:

```
[root@curso8 ~]#  named-checkconf /etc/named.conf
```

Se a execução do comando não retornar qualquer resultado, significa que as configurações estão corretas.

Para aplicar essas configurações, utilizamos o comando **rndc** junto ao subcomando **reconfig**:

```
[root@curso8 ~]#  rndc reconfig
```

Importante:

O subcomando **reconfig** é responsável por recarregar o arquivo de configuração do **named** e as novas zonas de **DNS** que tenham sido criadas – mas não recarrega aquelas existentes (mesmo se tiverem sido alteradas). Para fazer com que essas últimas também sejam recarregadas, utilizamos o subcomando **reload**.

O comando **kill** também poderá ser utilizado para fazer com que o daemon releia os seus arquivos de configuração:

```
[root@curso8 ~]#  kill -s HUP $(cat /run/named/named.pid).
```

O arquivo **/run/named/named.pid** é reponsável por registrar o **PID** relacionado ao daemon.

Vamos agora utilizar outro host para aplicar consultas ao **nameserver** do LAP1:

```
[root@curso8 ~]#  nslookup kali.org 192.168.0.15

Server:         192.168.0.15
Address:        192.168.0.15#53

Non-authoritative answer:
Name:   kali.org
Address: 192.124.249.10
```

Voltando ao LAP1, vamos editar novamente o arquivo **/etc/named.conf** e configurar a instrução **recursion** como "no".

Após salvar a alteração e aplicá-la, observe o resultado do comando **host** executado no próprio sistema:

```
[root@curso8 ~]#  host kali.org

Host kali.org not found: 5(REFUSED)
```

Como agora o **name server** não consegue manipular **consultas recursivas**, ele não tem como saber onde procurar informações sobre o domínio que lhe fôra informado.

Esta é a importância da instrução **recursion**. Ou seja, nesse momento, este **nameserver** somente conseguirá oferecer informações sobre zonas que por ele sejam manipuladas:

```
[root@curso8 ~]#  host localhost

localhost has address 127.0.0.1
localhost has IPv6 address ::1
```

Veja que nesse caso o procedimento de resolução de nomes foi realizado com sucesso. Por quê?

Como podemos visualizar no arquivo **/etc/named.conf**, o conteúdo do arquivo **/etc/named.rfc1912.zones** é carregado ao daemon **named** a partir da instrução **include**.

Se analisarmos o conteúdo deste arquivo, veremos que, dentre as diversas configurações de zonas locais, lá encontra-se a definição da zona conhecida como "localhost":

```
zone "localhost" IN {
type master;
      file "named.localhost";
      allow-update { none; };
};
```

Veja que o tipo desta zona está configurado como **master**, indicando que o arquivo de configuração desta zona encontra-se neste sistema (é manipulado diretamente por ele). Este arquivo é indicado pela instrução **file**, denominado como "named.localhost".

Como **directory** está configurada como **/var/named**, este arquivo localiza-se sob este diretório. Observe o seu conteúdo:

```
$TTL 1D
@       IN SOA  @ rname.invalid. (
0       ; serial
1D      ; refresh
1H      ; retry
1W      ; expire
3H )    ; minimum

NS    @
A       127.0.0.1
AAAA    ::1
```

Desta forma, sobre esta zona, por exemplo, este **nameserver** atua como um **master**. Por enquanto, memorize apenas a estrutura visual deste arquivo. Ao trabalharmos com zonas mais a frente entederemos cada um desses elementos.

Vamos configurar novamente a instrução **recursion** para "yes" e aplicar as alterações através do comando **rndc** junto ao **reconfig**.

2.4 Manipulação do Subsistema de Logs do BIND

O subsistema de log do **BIND** é controlado pelo bloco de instrução **logging** presente no arquivo **/etc/named.conf**:

```
logging {
channel default_debug {
            file "data/named.run";
            severity dynamic;
};
};
```

Dentro deste bloco poderemos encontrar diferentes instruções **channel**, já que todos os logs do **BIND** são manipuláveis através do uso de canais. Cada um deles é responsável por definir aspectos importantes da manipulação de logs do **BIND**, como o destino que as mensagens terão.

Por exemplo, no canal visto acima (denominado "default_debug"), as mensagens são encaminhadas ao arquivo **named.run**, presente em **/var/named/data**.

O canal também permite determinar a severidade (importância) das mensagens que serão manipuladas. O **BIND** classifica esses níveis em 7 tipos, listados a seguir (Os mais importantes são exibidos primeiro):

- **critical**
- **error**
- **warning**
- **notice**
- **info**
- **debug**
- **dynamic**

A severidade **debug** consegue trabalhar com níveis de detalhamento diferentes (1, 2...). O padrão a ser usado é o valor 1. Já a severidade **dynamic** é similar a anterior, porém utiliza quaisquer níveis de "debug" quando o **BIND** é inicializado.

Após definir o canal, é preciso estabelecer quais as categorias de dados a serem manipulados por este. O **BIND** disponibiliza diversas categorias, como "queries", "security", "notify" dentre outras.

Para entendermos isso tudo na prática, vamos editar o arquivo **/etc/named.conf** e incluir a seguinte definição de canal dentro da seção **logging**:

```
channel meus-registros {
file "data/meus-registros.txt";
severity debug;
};
```

Veja que criamos o canal "meus-registros" e que o arquivo relacionado a ele deverá estar posicionado sob **/var/named/data/**, denominado "meus-registros.txt".

Determinamos ainda que as mensagens que lá serão gravadas estarão associadas ao nível de severidade **debug**.

Agora vamos especificar a categoria das mensagens a serem associadas a este canal. Nesse exemplo, desejamos registrar eventos de segurança e consultas diversas que são aplicadas ao **nameserver**:

```
logging {
channel default_debug {
file "data/named.run";
            severity dynamic;
      };
      channel meus-registros {
            file "data/meus-registros.txt";
            severity debug;
      };
      category queries { meus-registros; };
      category security { meus-registros; default_debug; };
};
```

Veja que mensagens do tipo "security", por exemplo, além de irem para o canal que criamos, também possui agora como destino o canal "default_debug" – que já se encontra definido no arquivo.

Após salvar as edições, antes de aplicá-las, é importante lançarmos novamente o comando **named-checkconf** para verificar se existem erros de sintaxe.

Caso esteja tudo OK, vamos utilizar o comando **systemctl** para reiniciar o serviço:

```
[root@curso8 ~]#  systemctl restart named
```

Percebemos que o daemon já cria automaticamente o arquivo "meus-registros.txt" sob o diretório **/var/named/data**:

```
[root@curso8 ~]#  ls -l /var/named/data

total 56
-rw-r--r-- 1 named named     0 Jan 16 14:00 meus-registros.txt
-rw-r--r-- 1 named named 57128 Jan 16 14:00 named.run
```

Para testarmos esta configuração que criamos, vamos criar uma consulta com o comando **dig** no próprio host local:

```
[root@curso8 ~]#  dig linux.com

; <<>> DiG 9.11.4-P2-RedHat-9.11.4-9.P2.el7 <<>> linux.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 728
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 4, ADDITIONAL: 9

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;linux.com.                     IN      A

;; ANSWER SECTION:
linux.com.              300     IN      A       23.185.0.3

;; AUTHORITY SECTION:
linux.com.              172799  IN      NS      ns2.dnsimple.com.
linux.com.              172799  IN      NS      ns3.dnsimple.com.
linux.com.              172799  IN      NS      ns1.dnsimple.com.
linux.com.              172799  IN      NS      ns4.dnsimple.com.

;; ADDITIONAL SECTION:
ns1.dnsimple.com.       172799  IN      A       162.159.24.4
ns2.dnsimple.com.       172799  IN      A       162.159.25.4
ns3.dnsimple.com.       172799  IN      A       162.159.26.4
ns4.dnsimple.com.       172799  IN      A       162.159.27.4
ns1.dnsimple.com.       172799  IN      AAAA    2400:cb00:2049:1::a29f:1804
ns2.dnsimple.com.       172799  IN      AAAA    2400:cb00:2049:1::a29f:1904
ns3.dnsimple.com.       172799  IN      AAAA    2400:cb00:2049:1::a29f:1a04
ns4.dnsimple.com.       172799  IN      AAAA    2400:cb00:2049:1::a29f:1b04

;; Query time: 1751 msec
;; SERVER: 127.0.0.1#53(127.0.0.1)
;; WHEN: Thu Jan 16 18:56:19 -03 2020
;; MSG SIZE  rcvd: 311
```

O registro referente a esta consulta já poderá ser verificada no arquivo de log:

```
[root@curso8 ~]#  cat /var/named/data/meus-registros.txt

client @0x7f821803c1f0 127.0.0.1#34479 (linux.com): query: linux.com IN A +E(0)
(127.0.0.1)
```

O **BIND** permite ainda refinarmos um pouco mais os registros a serem gravados. Vamos até o arquivo **/etc/named.conf** e incluir as instruções **print-time** e **print-category**:

```
channel meus-registros {
file "data/meus-registros.txt";
      severity debug;
      print-time yes;
      print-category yes;
};
```

Essas instruções são utilizadas para que sejam gravados o momento em que consulta foi lançada e a categoria correspondente ao registro, respectivamente.

Vamos então remover o arquivo **/var/named/data/meus-registros.txt** e recarregar o daemon **named**:

```
[root@curso8 ~]#  rm -f /var/named/data/meus-registros.txt

[root@curso8 ~]#  rndc reconfig
```

Após lançarmos outra consulta, podemos visualizar no arquivo de log as novas instruções que aplicamos:

```
[root@curso8 ~]#  nslookup www.google.com

Server:         127.0.0.1
Address:        127.0.0.1#53

Non-authoritative answer:
Name:   www.google.com
Address: 172.217.29.68
Name:   www.google.com
Address: 2800:3f0:4004:807::2004

[root@curso8 ~]#  cat /var/named/data/meus-registros.txt

16-Jan-2020  18:58:47.416  queries:  client  @0x7f82180fff70  127.0.0.1#39570
(www.google.com): query: www.google.com IN A + (127.0.0.1)
16-Jan-2020  18:58:48.099  queries:  client  @0x7f821803c1f0  127.0.0.1#38338
(www.google.com): query: www.google.com IN AAAA + (127.0.0.1)
```

Para pararmos a execução do daemon **named**, o comando **rndc** poderá ser utilizado junto ao subcomando **stop**:

```
[root@curso8 ~]#  rndc stop
```

Entretanto, os comandos **systemctl**, **service** ou **kill** também poderão ser utilizados:

```
[root@curso8 ~]#  systemctl stop named

[root@curso8 ~]#  service named stop

[root@curso8 ~]#  kill -s SIGTERM $(cat /run/named/named.pid)
```

3 Criar e Manter Zonas de DNS

3.1 Introdução

Subtópico: 207.2 Peso: 3

Descrição do objetivo: Você deverá conseguir criar um arquivo de zona para redirecionamento, zona reversa ou hints para servidores de nível raiz. Este objetivo inclui a configuração apropriada de valores para registros, inclusão de hosts em zonas e inclusão de zonas ao DNS. Você também deverá conseguir delegar zonas para outro servidor de DNS.

Áreas-chave:

- Arquivos de configuração, termos e utilitários do BIND 9.
- Utilitários para requisitar informações a partir de um servidor de DNS.
- Layout, conteúdo e localização dos arquivos de zona do BIND.
- Vários métodos para adicionar um novo host nos arquivos de zona, incluindo zonas reversas.

Principais termos, arquivos e utilidades:

- /var/named/
- Sintaxe do arquivo de zona
- Formatos de registros de recursos
- named-checkzone
- named-compilezone
- masterfile-format
- dig
- nslookup
- host

3.2 Determinação da Zona Master

Vamos iniciar esta seção aprendendo sobre o processo de criação de uma zona de **DNS**. Para isto, utilizaremos o LAP1. Em seu arquivo **/etc/named.conf**, vamos estabelecer as seguintes configurações:

```
zone "dlteclpic2.com" IN {
type master;
      file "dlteclpic2.com";
      notify yes;
      allow-update { none; };
      allow-transfer { 192.168.0.16; };
};
```

Como este host irá manipular os dados originais da zona, denominada como "dlteclpic2.com", determinamos o seu tipo como **master**.

Além disso, configuramos a instrução **notify** como "yes", de forma que o **nameserver** a estar atuando como **Slave** nesta zona seja notificado sobre eventuais mudanças a ela aplicadas. Caso outros hosts também precisassem ser notificados, a instrução **also-notify** poderia ser utilizada.

Outra instrução importante é a **allow-update**, responsável por indicar quais hosts poderão submeter atualizações dinâmicas a esta zona. Como neste exemplo utilizamos a string "none", isto indica que nenhum host poderá realizar este procedimento.

Por fim, a instrução **allow-transfer** é responsável por indicar quais hosts estão autorizados a transferir (copiar) informações sobre esta zona. Neste exemplo, indicamos que somente o host de IP 192.168.0.16 encontra-se autorizado. É o **nameserver** deste último que iremos depois configurar para atuar como **Slave** na zona.

Além disso, também não será necessário que este **nameserver** atue com consultas recursivas. Sendo assim, alteramos a definição da instrução **recursion**:

```
recursion no;
```

Também podemos determinar que somente o sistema local ou o host 192.168.0.16 poderão efetuar consultas a ele:

```
allow-query { localhost; 192.168.0.16; };
```

Após salvar e sair do arquivo, não esqueça de aplicar o comando **named-checkconf** para verificar se existem erros de sintaxe.

Estando tudo OK, vamos ao diretório **/var/named** para criar o arquivo das configurações a serem utilizadas por esta zona.

Este arquivo, nomeado como "dlteclpic2.com" possuirá as seguintes configurações:

```
$TTL 604800
$ORIGIN dlteclpic2.com.

; Início do Bloco SOA

@       IN      SOA     lap1.dlteclpic2.com.    contato.dlteclpic2.com. (
1               ;       número do serial
10m             ;       verificar atualização
2m              ;       realizar nova tentativa
1W              ;       expiração das informações
5m              ;       cache de respostas negativas
)

; Especificação dos Name Servers atuantes na zona (Registros NS) e dos seus IP
; (Registros A)

@                   IN          NS          lap1.dlteclpic2.com.
lap1                IN          A           192.168.0.15

@                   IN          NS          lap2.dlteclpic2.com.
lap2                IN          A           192.168.0.16

; Especificação dos Servidores de E-mail da Zona (Registros MX) e seus IPs
;(Registros A)

@                   IN          MX      0   lap3.dlteclpic2.com.
lap3                IN          A           192.168.0.17

@                   IN          MX      10  lap4.dlteclpic2.com.
lap4                IN          A           192.168.0.18

; Especificação do Nome Canônico a ser dado ao host lap1 (Registro CNAME)

www                 IN          CNAME       lap1
```

Qualquer texto ou caractere posicionado após ";" é considerado como comentário do arquivo, ou seja, não faz parte das configurações a serem aplicadas à zona. Dito isto, vamos entender como funciona a lógica contida neste arquivo:

$TTL: esta instrução determina o Time To Live da zona – quanto tempo os resultados de uma consulta poderão permanecer na área de cache do **nameserver**. Estourado este tempo, uma nova pesquisa deverá ser realizada. Neste exemplo, estabelecemos que este tempo deverá ser de 86400 segundos (1 dia).

$ORIGIN: esta instrução determina o nome de domínio a ser utilizado quando esta informação não for declarada junto a um registro. Sendo assim, "lap1", por exemplo, será interpretado como "lap1.dlteclpic2.com.".

Caso a instrução não seja usada, o **BIND** sintetiza uma **$ORIGIN** a partir do nome da zona. Dessa forma, por exemplo, se o nome de uma zona é "exemplo.com" e no arquivo de definição da zona a instrução não for utilizada, esta é assumida com o mesmo nome da zona.

Em seguida entram em cena os registros dos recursos referente ao **SOA** (Start Of Authority). Vamos destrinchar primeiramente a linha que precede o início do seu bloco:

```
@       IN      SOA  lap1.dlteclpic2.com.    contato.dlteclpic2.com.
```

@: representa uma espécie de atalho para o nome contido na instrução **$ORIGIN** ou que tenha sido sintetizado pelo **BIND**.

IN: Indica que se trata de um registro da classe Internet.

SOA: Indica que esta é uma seção referente ao **SOA**.

lap1.dlteclpic2.com.: Indica que "lap1" é o nome da máquina cujo **nameserver** manipula o arquivo **master** referente a esta zona.

contato.dlteclpic2.com.: Indica o endereço de e-mail da pessoa que deverá ser contactada em casos de problemas relacionados a este domínio. Note que o caractere "." é utilizado após a string "contato" (ao invés do tradicional "@").

Logo em seguida tem início o bloco **SOA**. Preste bastante atenção na finalidade de cada uma das instruções e memorize a sua ordem:

1: indica o número de **serial** da zona. A cada mudança que seja aplicada às definições da zona, este número deverá crescer em 1, para que essas mudanças sejam noticiadas. Entenderemos melhor sobre isto mais a frente.

10m: indica o tempo de **refresh**, ou seja, de quanto em quanto tempo o **nameserver Slave** deverá verificar se houveram mudanças nas definições da zona. Neste exemplo, indicamos que este tempo será de 10 minutos.

2m: indica o tempo de **retry**, ou seja, quanto tempo o **nameserver Slave** deverá esperar para que uma nova tentativa de atualização dos dados desta zona seja iniciada novamente, após o processo ter sido interrompido por algum motivo. Neste exemplo, indicamos que este tempo será de 2 minutos.

1W: indica o tempo de **expire** dos dados da zona no lado "slave". Se um **nameserver** atuante como **Slave** nesta zona não contactar aquele atuante como **Master** para atualizar as suas informações em 1 semana (estabelecido neste exemplo), seus dados sobre a zona tornam-se expirados.

5m: indica o tempo de **negative caching**, ou seja, por quanto tempo o **nameserver** irá se lembrar de que não conseguiu resolver uma determinada consulta. Neste exemplo, indicamos que aquelas consultas não-resolvidas irão permanecer em cache por 5 minutos.

Quando o bloco **SOA** é finalizado, entram em cena as definições dos registros de recursos do tipo **NS**, ou seja, posicionamos aqueles **nameservers** que irão atuar sobre a zona (incluindo os seu respectivos IPs – associado pelo tipo de registro **A**):

```
@       IN      NS      lap1.dlteclpic2.com.
lap1    IN      A       192.168.0.15

@       IN      NS      lap2.dlteclpic2.com.
lap2    IN      A       192.168.0.16
```

Caso endereços IPv6 fossem utilizados, o tipo de registro de recursos seria o **AAAA**.

Em seguida as definições dos registros de recursos do tipo **MX** (Mail Exchange) são especificadas, responsáveis por indicar os servidores de e-mail a serem utilizados na zona. Além disso, os seus respectivos IPs também são configurados:

```
@       IN      MX      0       lap3.dlteclpic2.com.
lap3    IN      A               192.168.0.17

@       IN      MX      10      lap4.dlteclpic2.com.
lap4    IN      A               192.168.0.18
```

Veja que, enquanto "lap3" possui o valor 0 associado, o "lap4" está associado ao valor 10. Esses valores determinam a prioridade que deverá ser levada em consideração a ambos. O valor 0 representa a mais alta prioridade. Ou seja, quanto maior o número indicado neste terceiro campo, menor será a prioridade.

Dito isto, aquele servidor de prioridade 0 será usado primariamente para tal finalidade. Caso este servidor de e-mail estiver indisponível, outro host com o menor valor de prioridade será usado.

Por fim, é especificado que "www" representará um nome canônico para o host "lap1". Esta instrução é realizada a partir do registro de recursos **CNAME**:

```
www     IN      CNAME   lap1
```

Neste exemplo, especificamos que "www" representará o nome que estará associado ao host LAP1.

Após salvar e sair da edição arquivo, podemos verificar se nele existem erros de sintaxe. O comando a ser utilizado nesses casos é o **named-checkzone**. Primeiro especificamos o domínio e, em seguida, o seu arquivo de zona:

```
[root@curso1 named]#  named-checkzone dlteclpic2.com. dlteclpic2.com

zone dlteclpic2.com/IN: loaded serial 1
OK
```

Como tudo está OK, lançamos o comando **rndc** junto ao subcomando **reconfig**.

Depois, a partir do host 192.168.0.16, lançamos o comando **dig** para obter informações a respeito do host "www" da zona configurada no LAP1:

```
[root@curso8 ~]#  dig www.dlteclpic2.com @192.168.0.15

; <<>> DiG 9.10.3-P4-Debian <<>> www.dlteclpic2.com @192.168.0.15
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 63250
;; flags: qr aa rd; QUERY: 1, ANSWER: 2, AUTHORITY: 2, ADDITIONAL: 2
;; WARNING: recursion requested but not available

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;www.dlteclpic2.com.            IN    A

;; ANSWER SECTION:
www.dlteclpic2.com.     604800      IN    CNAME lap1.dlteclpic2.com.
lap1.dlteclpic2.com.    604800      IN    A     192.168.0.15

;; AUTHORITY SECTION:
dlteclpic2.com.         604800      IN    NS    lap1.dlteclpic2.com.
dlteclpic2.com.         604800      IN    NS    lap2.dlteclpic2.com.

;; ADDITIONAL SECTION:
lap2.dlteclpic2.com.    604800      IN    A     192.168.0.16

;; Query time: 1 msec
;; SERVER: 192.168.0.15#53(192.168.0.15)
;; WHEN: Thu Jan 16 22:08:36 EST 2020
;; MSG SIZE  rcvd: 131
```

Para visualizarmos informações a respeito dos servidores de e-mail utilizados na zona, especificamos ao **dig** o tipo de consulta **MX**:

```
[root@curso8 ~]#  dig dlteclpic2.com @192.168.0.15  MX

; <<>> DiG 9.10.3-P4-Debian <<>> dlteclpic2.com @192.168.0.15 MX
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 26237
;; flags: qr aa rd; QUERY: 1, ANSWER: 2, AUTHORITY: 2, ADDITIONAL: 5
;; WARNING: recursion requested but not available

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;dlteclpic2.com.                IN    MX

;; ANSWER SECTION:
dlteclpic2.com.         604800      IN    MX    0 lap3.dlteclpic2.com.
dlteclpic2.com.         604800      IN    MX    10 lap4.dlteclpic2.com.

;; AUTHORITY SECTION:
dlteclpic2.com.         604800      IN    NS    lap1.dlteclpic2.com.
dlteclpic2.com.         604800      IN    NS    lap2.dlteclpic2.com.
```

```
;; ADDITIONAL SECTION:
lap3.dlteclpic2.com.    604800      IN    A     192.168.0.17
lap4.dlteclpic2.com.    604800      IN    A     192.168.0.18
lap1.dlteclpic2.com.    604800      IN    A     192.168.0.15
lap2.dlteclpic2.com.    604800      IN    A     192.168.0.16

;; Query time: 0 msec
;; SERVER: 192.168.0.15#53(192.168.0.15)
;; WHEN: Thu Jan 16 22:09:15 EST 2020
;; MSG SIZE  rcvd: 187
```

O comando **host** também permite visualizar informações de forma bastante objetiva:

```
[root@curso8 ~]#  host www.dlteclpic2.com 192.168.0.15

Using domain server:
Name: 192.168.0.15
Address: 192.168.0.15#53
Aliases:

www.dlteclpic2.com is an alias for lap1.dlteclpic2.com.
lap1.dlteclpic2.com has address 192.168.0.15
```

O **nslookup** também permite visualizar esta mesma informação, mas de forma um pouco diferente:

```
[root@curso8 ~]#  nslookup www.dlteclpic2.com 192.168.0.15

Server:           192.168.0.15
Address:    192.168.0.15#53

www.dlteclpic2.com      canonical name = lap1.dlteclpic2.com.
Name: lap1.dlteclpic2.com
Address: 192.168.0.15
```

3.3 Determinação da Zona Slave

Estando agora no host LAP2 (IP: 192.168.0.16) com o Debian 9.11 em execução, vamos configurá-lo agora para atuar como **Slave** na zona que criamos. O pacote responsável por instalar o **BIND** em distribuições baseadas no Debian é identificado como "bind9". Além disso, outro pacote também deverá ser instalado – o "bind9utils".

Para configurarmos este host para atuar como **Slave**, precisamos incluir a definição da zona no arquivo **/etc/bind/named.conf.local**:

```
zone "dlteclpic2.com" IN {
      type slave;
      masters { 192.168.0.15; };
      file "dlteclpic2.com";
};
```

Note a importância da instrução **masters** contida na definição da zona. É através dela que podemos especificar o endereço IP do LAP1.

Para finalizar o procedimento, após salvar o arquivo, vamos iniciar o daemon do **BIND**. Nesses cenários, o daemon do serviço é identificado como **bind9**.

```
[root@curso8 ~]#  systemctl start bind9
```

Em seguida, podemos visualizar o conteúdo do arquivo **/var/log/syslog** e perceber que o processo de transferência das informações da zona contidas em LAP1 já foram recebidas pelo host local:

```
[root@curso8 ~]#  tail /var/log/syslog

Jan 16 22:13:00 lap2 named[2542]: running
Jan 16 22:13:00 lap2 named[2542]: network unreachable resolving './DNSKEY/IN':
2001:500:84::b#53
Jan 16 22:13:00 lap2 named[2542]: network unreachable resolving './NS/IN':
2001:500:84::b#53
Jan 16 22:13:00 lap2 named[2542]: zone dlteclpic2.com/IN: Transfer started.
Jan 16 22:13:00 lap2 named[2542]: transfer of 'dlteclpic2.com/IN' from
192.168.0.15#53: connected using 192.168.0.16#47113
Jan 16 22:13:00 lap2 named[2542]: zone dlteclpic2.com/IN: transferred serial 1
Jan 16 22:13:00 lap2 named[2542]: transfer of 'dlteclpic2.com/IN' from
192.168.0.15#53: Transfer status: success
Jan 16 22:13:00 lap2 named[2542]: transfer of 'dlteclpic2.com/IN' from
192.168.0.15#53: Transfer completed: 1 messages, 11 records, 274 bytes, 0.001
secs (274000 bytes/sec)
Jan 16 22:13:00 lap2 named[2542]: zone dlteclpic2.com/IN: sending notifies
(serial 1)
```

Estando no diretório **/var/cache/bind**, podemos visualizar o devido posicionamento do arquivo referente a zona em questão:

```
-rw-r--r-- 1 bind bind 546 Jan 15 17:11 dlteclpic2.com
```

Entretanto este arquivo não poderá ser visualizado de forma direta, já que encontra-se em um formato que denominamos como **raw**. Para gerar uma versão **text** deste, utilizamos o comando **named-compilezone**:

```
[root@curso8 /var/cache/bind]#  named-compilezone -f raw -F text -o

dlteclpic2.com.txt dlteclpic2.com dlteclpic2.com
zone dlteclpic2.com/IN: loaded serial 1
dump zone to dlteclpic2.com.txt...done
OK
```

A opção **–f** é utilizado para que seja especificado o formato de entrada do arquivo, e a opção **–F** permite especificar o formato que o arquivo terá como saída. A opção **–o** permite especificar o nome do arquivo a ser gerado.

Para fechar a sequência do comando, especificamos o domínio (dlteclpic2.com) e, em seguida, o arquivo de zona que está sendo usado como base para o procedimento (dlteclpic2.com).

Agora podemos lançar o comando **cat** para visualizarmos o conteúdo do arquivo que geramos:

```
[root@curso1:/var/cache/bind]#  cat dlteclpic2.com.txt

dlteclpic2.com.                         604800 IN   SOA   lap1.dlteclpic2.com.
contato.dlteclpic2.com. 1 600 120 604800 300
dlteclpic2.com.                         604800 IN   NS    lap1.dlteclpic2.com.
dlteclpic2.com.                         604800 IN   NS    lap2.dlteclpic2.com.
dlteclpic2.com.                         604800 IN   MX    0
lap3.dlteclpic2.com.
dlteclpic2.com.                         604800 IN   MX    10
lap4.dlteclpic2.com.
lap1.dlteclpic2.com.                    604800 IN   A     192.168.0.15
lap2.dlteclpic2.com.                    604800 IN   A     192.168.0.16
lap3.dlteclpic2.com.                    604800 IN   A     192.168.0.17
lap4.dlteclpic2.com.                    604800 IN   A     192.168.0.18
www.dlteclpic2.com.                     604800 IN   CNAME lap1.dlteclpic2.com.
```

Estando ainda no mesmo host, vamos lançar o comando **dig** apenas para testar o retorno oferecido pelo comando:

```
[root@curso8 ~]#  dig www.dlteclpic2.com @192.168.0.15

; <<>> DiG 9.10.3-P4-Debian <<>> www.dlteclpic2.com @192.168.0.15
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 2904
;; flags: qr aa rd; QUERY: 1, ANSWER: 2, AUTHORITY: 2, ADDITIONAL: 2
;; WARNING: recursion requested but not available

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;www.dlteclpic2.com.            IN    A

;; ANSWER SECTION:
www.dlteclpic2.com.     604800      IN    CNAME lap1.dlteclpic2.com.
lap1.dlteclpic2.com.    604800      IN    A     192.168.0.15

;; AUTHORITY SECTION:
```

```
dlteclpic2.com.         604800      IN    NS    lap1.dlteclpic2.com.
dlteclpic2.com.         604800      IN    NS    lap2.dlteclpic2.com.

;; ADDITIONAL SECTION:
lap2.dlteclpic2.com.    604800      IN    A     192.168.0.16

;; Query time: 0 msec
;; SERVER: 192.168.0.15#53(192.168.0.15)
;; WHEN: Thu Jan 16 22:22:39 EST 2020
;; MSG SIZE  rcvd: 131
```

Para finalizarmos esta seção do capítulo, vamos retornar ao LAP1 e alterar o serial do arquivo **/var/named/dlteclpic2.com** para 2.

Como alteramos um arquivo de zona que já existia, podemos lancer o comando **rndc** junto ao subcomando **reload** para que os arquivos do daemon **named** sejam recarregados:

```
[root@curso8 ~]#  rndc reload

server reload successful
```

Retornando ao LAP2, podemos visualizar que o arquivo **/var/log/syslog** já registra o processo de transferência de zonas – já que o serial da base original foi alterado:

```
[root@curso8 ~]#  tail /var/log/syslog

Jan 16 22:24:46 lap2 named[2542]: client 192.168.0.15#59165: received notify for
zone 'dlteclpic2.com'
Jan 16 22:24:46 lap2 named[2542]: zone dlteclpic2.com/IN: notify from
192.168.0.15#59165: serial 2
Jan 16 22:24:46 lap3 named[2542]: zone dlteclpic2.com/IN: Transfer started.
Jan 16 22:24:46 lap2 named[2542]: transfer of 'dlteclpic2.com/IN' from
192.168.0.15#53: connected using 192.168.0.16#36775
Jan 16 22:24:46 lap3 named[2542]: zone dlteclpic2.com/IN: transferred serial 2
Jan 16 22:24:46 lap2 named[2542]: transfer of 'dlteclpic2.com/IN' from
192.168.0.15#53: Transfer status: success
Jan 16 22:24:46 lap2 named[2542]: transfer of 'dlteclpic2.com/IN' from
192.168.0.15#53: Transfer completed: 1 messages, 11 records, 274 bytes, 0.001
secs (274000 bytes/sec)
Jan 16 22:24:46 lap2 named[2542]: zone dlteclpic2.com/IN: sending notifies
(serial 2)
```

3.4 Zona Reversa e Encaminhamento de Queries

A zona reversa é aquela que possui a capacidade de mapear um endereço **IP** a um **FQDN**. Seu nome é composto pela identificação da rede ao contrário incluindo a string "in-addr.arpa".

Estando no LAP1, vamos editar o arquivo **/etc/named.conf** e incluir a seguinte configuração:

```
zone "0.168.192.in-addr.arpa" IN {
      type master;
      file "dlteclpic2-reverso.com";
      notify yes;
      allow-update { none; };
};
```

Após salvar o arquivo e lançar o comando **named-checkconf** para verificar a sua sintaxe, vamos até o diretório **/var/named** e definir o arquivo "dlteclpic2-reverso.com":

```
$TTL 1800
$ORIGIN 0.168.192.in-addr.arpa.

@       IN      SOA     lap1.dlteclpic2.com.    contato.dlteclpic2.com. (
0               ;       número do serial
10m             ;       verificar atualização
2m              ;       realizar nova tentativa
1W              ;       expiração das informações
5m              ;       cache de respostas negativas
)

        IN        NS    lap1.dlteclpic2.com.
        IN        NS    lap2.dlteclpic2.com.

15      IN      PTR     lap1.dlteclpic2.com.
16      IN      PTR     lap2.dlteclpic2.com.
17      IN      PTR     lap3.dlteclpic2.com.
18      IN      PTR     lap4.dlteclpic2.com.
```

Note que utilizamos a classe de registro **PTR** para trabalhar com este tipo de zona. Este registro é responsável por criar a associação entre os endereços IP da zona reversa com os nomes de hosts. Desta forma, por exemplo, "lap1.dlteclpic2.com". será mapeado ao endereço 192.168.0.15.

Após salvar o arquivo, vamos aplicar o comando **named-checkzone** para analisar a sua sintaxe:

```
[root@curso8 named]# named-checkzone 0.168.192.in-addr.arpa dlteclpic2-
reverso.com

zone 0.168.192.in-addr.arpa/IN: loaded serial 0
OK
```

Para finalizar este procedimento, lançamos o comando **rndc** junto ao subcomando **reconfig**. Agora vamos até o LAP2 e, após parar o daemon **bind9**, editamos o arquivo **/etc/bind/named.conf.local** com o seguinte conteúdo:

```
zone "0.168.192.in-addr.arpa" IN {
      type slave;
      file "dlteclpic2-reverso.com";
      masters { 192.168.0.15; };
};
```

Em seguida, lançamos o comando **systemctl** para iniciar novamente o daemon:

```
[root@curso8 ~]#  systemctl start bind9
```

Ainda no LAP2, vamos realizar alguns testes. Acompanhe:

```
[root@curso8 ~]#  host 192.168.0.18 192.168.0.15

Using domain server:
Name: 192.168.0.15
Address: 192.168.0.15#53
Aliases:

18.0.168.192.in-addr.arpa domain name pointer lap4.dlteclpic2.com.

[root@curso8 ~]#  nslookup 192.168.0.17 192.168.0.15

Server:           192.168.0.15
Address:    192.168.0.15#53

17.0.168.192.in-addr.arpa     name = lap3.dlteclpic2.com.

[root@curso8 ~]#  dig +short -x 192.168.0.17 @192.168.0.15

lap3.dlteclpic2.com.
```

Um **name server** também consegue atuar como um encaminhador de consultas para uma determinada zona, bastando para isto defini-la com o tipo **forward**.

Para compor os próximos exemplos, utilizaremos um host com IP é 192.168.0.200, rodando o CentOS. Seu hostname é HOST1.

Dito isto, em seu arquivo **/etc/named.conf**, configuramos a zona da seguinte forma:

```
zone "dlteclpic2.com" IN {
      type forward;
      forwarders { 192.168.0.15; 192.168.0.16; };
      forward only;

};

zone "0.168.192.in-addr.arpa" IN {
      type forward;
      forwarders { 192.168.0.15; 192.168.0.16; };
      forward only;
};
```

Perceba que os endereços IP do LAP1 e do LAP2 são discriminados na instrução **forwarders**. Repare também que na definição da zona reversa utilizamos a instrução **forward**, especificando-lhe a opção "only". Isso indica que este **nameserver** apenas irá encaminhar as consultas a tais zonas.

Em seguida, vamos configurar a sua seção **options** da seguinte forma:

```
options {
listen-on port 53 { 127.0.0.1; 192.168.0.200; };
      listen-on-v6 port 53 { ::1; };
      directory       "/var/named";
      dump-file       "/var/named/data/cache_dump.db";
      statistics-file "/var/named/data/named_stats.txt";
      memstatistics-file "/var/named/data/named_mem_stats.txt";
      recursing-file  "/var/named/data/named.recursing";
      secroots-file   "/var/named/data/named.secroots";
      allow-query     { localhost; autorizados; };
recursion yes;
dnssec-enable no;
      dnssec-validation no;
bindkeys-file "/etc/named.root.key";
managed-keys-directory "/var/named/dynamic";
pid-file "/run/named/named.pid";
      session-keyfile "/run/named/session.key";
      empty-zones-enable no;
};
```

Perceba que na instrução **allow-query** foi especificada a string "autorizados". Para isso, iremos criar a **acl** correspondente fora desta seção:

```
acl autorizados { localhost; localnets; };
```

Desta forma, estamos autorizando ao localhost e demais hosts presentes na rede local (localnets) para que consigam efetuar consultas recursivas a este **name server**. Mais a frente estudaremos sobre o **DNSSEC**. Portanto, por hora, configuramos as instruções **dnssec-enable** e **dnssec-validation** como "no".

Note ainda que configuramos a instrução **empty-zones-enable** como "no", de forma a permitir que criemos um redirecionamento específico para a zona reversa.

Após salvar o arquivo, verificar sua sintaxe com o **named-checkconf** e executar o comando **rndc** junto ao **reconfig**, vamos até o LAP1. Lá iremos editar a instrução **allow-query** para incluir o IP do HOST1:

```
allow-query { localhost; 192.168.0.16; 192.168.0.200 };
```

Após salvar a alteração e executar o comando **rndc** junto ao subcomando **reconfig**, vamos utilizar um outro host da rede para lançar consultas ao HOST1, objetivando respostas a respeito da zona "dlteclpic2.com":

```
[root@curso8 ~]#  dig www.dlteclpic2.com @192.168.0.200

; <<>> DiG 9.8.1-P1 <<>> www.dlteclpic2.com @192.168.0.200
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 62501
;; flags: qr rd ra; QUERY: 1, ANSWER: 2, AUTHORITY: 2, ADDITIONAL: 1

;; QUESTION SECTION:
;www.dlteclpic2.com.            IN    A

;; ANSWER SECTION:
www.dlteclpic2.com.     604800      IN    CNAME lap1.dlteclpic2.com.
lap1.dlteclpic2.com.    604800      IN    A     192.168.0.15

;; AUTHORITY SECTION:
dlteclpic2.com.         604800      IN    NS    lap1.dlteclpic2.com.
dlteclpic2.com.         604800      IN    NS    lap2.dlteclpic2.com.

;; ADDITIONAL SECTION:
lap2.dlteclpic2.com.    604800      IN    A     192.168.0.16

;; Query time: 1733 msec
;; SERVER: 192.168.0.200#53(192.168.0.200)
;; WHEN: Fri Jan 17 02:25:23 2020
;; MSG SIZE  rcvd: 120


[root@curso8 ~]#  nslookup lap3.dlteclpic2.com 192.168.0.200

Server:           192.168.0.200
Address:    192.168.0.200#53

Non-authoritative answer:
Name: lap3.dlteclpic2.com
Address: 192.168.0.17

[root@curso1 ~]#  host 192.168.0.18 192.168.0.200

Using domain server:
Name: 192.168.0.200
Address: 192.168.0.200#53
Aliases:

18.0.168.192.in-addr.arpa domain name pointer lap4.dlteclpic2.com.
```

Retornando ao HOST1, vamos agora criar um redirecionamento para ser utilizado de forma externa, ou seja, iremos especificar **name servers** que deverão ser acionados para realizarem o processo de resolução de nomes a uma dada consulta. Para isto, em **/etc/named.conf**, incluímos a instrução **forwarders** na seção **options**:

```
options {
listen-on port 53 { 127.0.0.1; 192.168.0.88; };
      listen-on-v6 port 53 { ::1; };
      directory       "/var/named";
      dump-file       "/var/named/data/cache_dump.db";
      statistics-file "/var/named/data/named_stats.txt";
      memstatistics-file "/var/named/data/named_mem_stats.txt";
      recursing-file  "/var/named/data/named.recursing";
      secroots-file   "/var/named/data/named.secroots";
      allow-query     { localhost; autorizados; };
recursion yes;
dnssec-enable no;
      dnssec-validation no;
bindkeys-file "/etc/named.root.key";
managed-keys-directory "/var/named/dynamic";
pid-file "/run/named/named.pid";
      session-keyfile "/run/named/session.key";
      empty-zones-enable no;
      forwarders { 8.8.8.8; 8.8.4.4; };
      forward only;
};
```

Após salvar o arquivo e aplicar o comando **named-checkconf** sobre para verificar sua sintaxe, utilizamos o comando **rndc** junto ao **reconfig** para aplicar tais configurações.

Utilizando novamente o outro host, podemos lançar (por exemplo) o comando **dig**:

```
[root@curso8 ~]#  dig www.mageia.com @192.168.0.200

; <<>> DiG 9.8.1-P1 <<>> www.mageia.com @192.168.0.200
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 24230
;; flags: qr rd ra; QUERY: 1, ANSWER: 2, AUTHORITY: 2, ADDITIONAL: 4

;; QUESTION SECTION:
;www.mageia.com.                IN    A

;; ANSWER SECTION:
www.mageia.com.         3600   IN    CNAME mageia.com.
mageia.com.        600  IN    A     50.62.198.70

;; AUTHORITY SECTION:
mageia.com.        3600   IN    NS    ns24.domaincontrol.com.
mageia.com.        3600   IN    NS    ns23.domaincontrol.com.

;; ADDITIONAL SECTION:
ns24.domaincontrol.com. 172800      IN    A     173.201.69.12
ns23.domaincontrol.com. 172800      IN    A     97.74.101.12
ns24.domaincontrol.com. 172800      IN    AAAA  2603:5:2251::c
ns23.domaincontrol.com. 172800      IN    AAAA  2603:5:2151::c

;; Query time: 1638 msec
```

```
;; SERVER: 192.168.0.200#53(192.168.0.200)
;; WHEN: Fri Jan 17 02:39:39 2020
;; MSG SIZE  rcvd: 202
```

4 Segurança do Servidor de DNS

4.1 Introdução

Subtópico: 207.3 Peso: 2

Descrição do objetivo: Você deverá conseguir configurar um servidor de DNS para ser executado por um usuário não-root e no ambiente chroot jail. Este objetivo inclui a troca segura de dados entre servidores de DNS.

Áreas-chave:

- Arquivos de configuração do BIND 9.
- Configuração do BIND para ser executado em um ambiente chroot jail.
- Dividir as configurações do BIND usando redirecionamentos.
- Configurar e utilizar assinaturas de transação (TSIG).
- Ciência sobre o DNSSEC e ferramentas básicas.
- Ciência sobre o DANE e registros relacionados.

Principais termos, arquivos e utilidades:

- /etc/named.conf
- /etc/passwd
- DNSSEC
- dnssec-keygen
- dnssec-signzone

4.2 Implementação de Medidas Básicas de Segurança

Diversos critérios sobre segurança deverão ser levados em consideração quando o assunto DNS vem à tona, já que trata-se de um elemento crítico e bastante visado por atacantes.

Não é incomum que áreas de cache de **nameservers** sejam manipuladas de maneira maliciosa, permitindo que consultas sejam redirecionadas a sites enganosos – método conhecido como **Cache Poisoning**.

O **LPI** cobra do candidato alguns critérios básicos e importantes relacionados a gestão segura de servidores de DNS utilizando o **BIND**. Porém, como trata-se de um assunto amplo e com uma vasta bibliografia, é recomendado que você se aprofunde e conheça mais sobre o assunto.

A medida mais elementar de todas é buscar sempre manter o **BIND** atualizado. Apesar de ser uma medida de segurança básica, não é raro encontrar servidores de **DNS** em execução na web apresentando versões antigas, contendo brechas de segurança ainda não resolvidas. Sua versão poderá ser conferida, por exemplo, através da saída do comando **named**:

```
[root@curso8 ~]# named -v

BIND 9.11.4-P2-RedHat-9.11.4-9.P2.el7 (Extended Support Version) <id:7107deb>
```

Por questões de segurança, não é recomendado disponibilizar esta informação a hosts externos. Por exemplo, estando em um host qualquer, vamos lançar os comandos **dig** e **nslookup** em direção ao HOST1:

```
[root@curso8 ~]# dig @192.168.0.200 -c CH -t txt version.bind

; <<>> DiG 9.8.1-P1 <<>> @192.168.0.200 -c CH -t txt version.bind
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 33880
;; flags: qr aa rd; QUERY: 1, ANSWER: 1, AUTHORITY: 1, ADDITIONAL: 0
;; WARNING: recursion requested but not available

;; QUESTION SECTION:
;version.bind.                  CH     TXT
```

```
;; ANSWER SECTION:
version.bind.           0       CH      TXT     "9.11.4-P2-RedHat-9.11.4-9.P2.el7"

;; AUTHORITY SECTION:
version.bind.           0       CH      NS      version.bind.

;; Query time: 40 msec
;; SERVER: 192.168.0.200#53(192.168.0.200)
;; WHEN: Fri Jan 17 09:27:16 2020
;; MSG SIZE  rcvd: 89


[root@curso8 ~]# nslookup -type=txt -class=chaos version.bind 192.168.0.21

Server:         192.168.0.200
Address:    192.168.0.200#53

version.bind    text = "9.11.4-P2-RedHat-9.11.4-9.P2.el7"
```

Veja que, utilizando a classe chaosnet, é possível identificar a versão do **BIND** em execução no HOST1. Para que esta informação não seja exibida, vamos até o HOST1 e, no arquivo **/etc/named.conf**, configuraremos a instrução **version** com a string "-- -- --", por exemplo:

```
options {
listen-on port 53 { 127.0.0.1; 192.168.0.200; };
      listen-on-v6 port 53 { ::1; };
      directory       "/var/named";
      dump-file       "/var/named/data/cache_dump.db";
      statistics-file "/var/named/data/named_stats.txt";
      memstatistics-file "/var/named/data/named_mem_stats.txt";
      recursing-file  "/var/named/data/named.recursing";
      secroots-file   "/var/named/data/named.secroots";
      allow-query     { localhost; autorizados; };
      recursion yes;
      dnssec-enable no;
      dnssec-validation no;
      bindkeys-file "/etc/named.root.key";
      managed-keys-directory "/var/named/dynamic";
      pid-file "/run/named/named.pid";
      session-keyfile "/run/named/session.key";
      empty-zones-enable no;
      forwarders { 8.8.8.8; 8.8.4.4; };
      forward only;
      version "--- --- ---";
};
```

Após verificar a sintaxe do arquivo com o comando **named-checkconf**, vamos executar o comando **rndc** junto ao **reconfig** para aplicar as alterações.

Retornando ao outro host, lançamos o comando **nslookup**, por exemplo, e vemos que a versão do **BIND** não é mais exibida:

```
[root@curso8 ~]# nslookup -type=txt -class=chaos version.bind 192.168.0.21

Server:           192.168.0.200
Address:    192.168.0.200#53

version.bind      text = "--- --- ---"
```

Outra medida de segurança importante é garantir que o host ofertante de serviços de **DNS** ofereça somente as funcionalidades que lhe são requeridas. Por exemplo, se o **nameserver** for utilizado como **Authoritative-only**, não é necessário habilitar o suporte a consultas recursivas.

Além disso, não é recomendado que o host utilizado para abrigar o **nameserver** seja compartilhado com outros serviços (web, bancos de dados etc), já que uma coisa pode interferir na outra.

As Transferências entre Zonas também deverão ser procedimentos bastante controlados. É possível utilizar o comando **dig** para iniciar este procedimento, bastando para isto especificar o tipo de consulta **axfr**.

Vamos partir do princípio que o arquivo **/etc/named.conf** do LAP1 possuísse as seguintes instruções presentes na seção **options**:

```
allow-query { any; }

allow-transfer { any; }
```

Vamos imaginar também que a definição da zona que criamos estivesse desta forma:

```
zone "dlteclpic2.com" IN {
        type master;
        file "dlteclpic2.com";
        notify yes;
        allow-update { none; };
};
```

Sendo assim, a partir de um host qualquer, lançamos o comando **dig** em direção ao LAP1:

```
[root@curso8 ~]# dig axfr dlteclpic2.com @192.168.0.15

; <<>> DiG 9.8.1-P1 <<>> axfr dlteclpic2.com @192.168.0.15
;; global options: +cmd
dlteclpic2.com.         604800      IN    SOA   lap1.dlteclpic2.com.
contato.dlteclpic2.com. 2 600 120 604800 300
dlteclpic2.com.         604800      IN    NS    lap1.dlteclpic2.com.
dlteclpic2.com.         604800      IN    NS    lap2.dlteclpic2.com.
dlteclpic2.com.         604800      IN    MX    0 lap3.dlteclpic2.com.
dlteclpic2.com.         604800      IN    MX    10 lap4.dlteclpic2.com.
lap1.dlteclpic2.com.    604800      IN    A     192.168.0.15
lap2.dlteclpic2.com.    604800      IN    A     192.168.0.16
lap3.dlteclpic2.com.    604800      IN    A     192.168.0.17
lap4.dlteclpic2.com.    604800      IN    A     192.168.0.18
www.dlteclpic2.com.     604800      IN    CNAME lap1.dlteclpic2.com.
dlteclpic2.com.         604800      IN    SOA   lap1.dlteclpic2.com.
contato.dlteclpic2.com. 2 600 120 604800 300
```

```
;; Query time: 3 msec
;; SERVER: 192.168.0.15#53(192.168.0.15)
;; WHEN: Sat Jan 18 10:35:39 2020
;; XFR size: 11 records (messages 1, bytes 274)
```

Veja que todas as informações a respeito da zona são então exibidas ao host que efetuou a consulta. O ideal é não permitir que isto aconteça (somente àqueles autorizados). Desta forma, nas definições da zona no LAP1, acrescentamos a instrução **allow-transfer**:

```
zone "dlteclpic2.com" IN {
        type master;
        file "dlteclpic2.com";
        notify yes;
        allow-update { none; };
        allow-transfer { 192.168.0.16; };
};
```

Sendo assim, agora especificamos que, para esta zona, somente serão permitidas transferências ao host 192.168.0.16 (LAP2). Por isso, se neste momento lançarmos o comando **dig** novamente a partir do outro host, essas informações da zona não serão mais exibidas.

Apesar do **allow-transfer** estar configurada como "any" na seção **options**, esta instrução específica na definição da zona é aquela a ser levada em consideração:

```
[root@curso8 ~]# dig axfr dlteclpic2.com @192.168.0.15

; <<>> DiG 9.8.1-P1 <<>> axfr dlteclpic2.com @192.168.0.15
;; global options: +cmd
; Transfer failed.
```

Já a instrução **blackhole** poderá também ser empregada para definir quais aqueles hosts que não poderão efetuar consultas ao nameserver. Observe agora a definição da instrução **allow-query** presente na seção options de LAP1:

```
allow-query { localhost; localnets; }
```

Isso quer dizer que, além do host local, qualquer outro presente na mesma rede que LAP1 conseguirá efetuar consultas ao seu **nameserver**. Porém, caso desejássemos especificar alguns endereços de forma a não permitir este procedimento, a instrução **blackhole** poderá ser especificada da seguinte forma:

```
allow-query { localhost; localnets; };

blackhole { 192.168.0.109; };
```

Desta forma, este **nameserver** irá recusar quaisquer requisições de consultas oriundas do host 192.168.0.200 (os demais da rede serão atendidos normalmente). Acompanhe:

```
# Consulta enviada pelo host 192.168.0.99

[root@curso8 ~]# dig +short www.dlteclpic2.com @192.168.0.15

lap1.dlteclpic2.com.
192.168.0.15

# Consulta enviada pelo host 192.168.0.109

[root@curso8 ~]# dig +short www.dlteclpic2.com @192.168.0.15

;; connection timed out; no servers could be reached
```

4.3 Propósito e Configuração de Views

O cenário conhecido como "Split Horizon" é aquele em que um **nameserver** possui a capacidade de oferecer resultados diferentes baseado no host que efetuou uma determinada consulta. Esse cenário é bastante comum naqueles ambientes em que um **nameserver** é utilizado para responder consultas internas e externas.

No **BIND**, obteremos esse cenário através do uso de **views** e **acl's**. Criaremos duas respostas diferentes para uma mesma zona. Para compor os próximos exemplos, vamos utilizar o HOST2, cujo IP é o 192.168.0.99 (com o CentOS rodando).

Sendo assim, em seu arquivo **/etc/named.conf**, vamos criar a seguinte **acl**:

```
acl autorizados {
        192.168.0.50; 192.168.0.51; 192.168.0.52;
};
```

Em seguida, no mesmo arquivo, vamos criar duas **views**:

```
view "visao-interna" {
        match-clients {autorizados; };
        include "/etc/named.interna.zones";
        zone "." IN {
                type hint;
                file "named.ca";
        };

        include "/etc/named.rfc1912.zones";
};

view "visao-externa" {
        match-clients { localhost; localnets; };
        include "/etc/named.externa.zones";
};
```

Agora vamos definir a zona "linuxstore.com" no arquivo **/etc/named.interna.zones**:

```
zone "linuxstore.com" IN {
        type master;
        file "/var/named/data/db_interna.linuxstore.com";
        allow-update { none; };
};
```

Vamos agora realizar a definição da zona no arquivo **/etc/named.externa.zones**:

```
zone "linuxstore.com" IN {
        type master;
        file "/var/named/data/db_externa.linuxstore.com";
        allow-update { none; };
};
```

Em seguida, vamos criar o arquivo **/var/named/data/db_interna.linuxstore.com** contendo dados a serem visualizados pelos hosts autorizados:

```
$TTL 1800
$ORIGIN linuxstore.com.

@       IN      SOA     maquina1.linuxstore.com.      admin.linuxstore.com. (
1
1D
1H
1W
3H
)

@                       IN      NS      maquina1.linuxstore.com.
maquina1                IN      A       192.168.0.15
@                       IN      NS      maquina2.linuxstore.com.
maquina2                IN      A       192.168.0.17
```

Agora vamos definir o arquivo **/var/named/data/db_externo.linuxstore.com**, a conter dados visualizáveis por qualquer host da rede:

```
$TTL 1800
$ORIGIN linuxstore.com.

@       IN      SOA     maquina1.linuxstore.com.      admin.linuxstore.com. (
1
1D
1H
1W
3H
)

@                       IN      NS      maquina1.linuxstore.com.
maquina1                IN      A       172.16.30.26
@                       IN      NS      maquina2.linuxstore.com.
maquina2                IN      A       172.16.30.27
```

Para finalizar o procedimento, vamos executar o comando **rndc** junto ao subcomando **reload**.

Em seguida, utilizando o host de IP 192.168.0.50, vamos lançar o comando **dig** em direção ao HOST2:

```
[root@curso8 ~]# dig @192.168.0.99 lap1.linuxstore.com

; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 49390
;; flags: qr aa rd; QUERY: 1, ANSWER: 1, AUTHORITY: 2, ADDITIONAL: 1
;; WARNING: recursion requested but not available

;; QUESTION SECTION:
;lap1.linuxstore.com.           IN      A

;; ANSWER SECTION:
lap1.linuxstore.com.    1800    IN      A       192.168.0.10

;; AUTHORITY SECTION:
linuxstore.com.         1800    IN      NS      lap2.linuxstore.com.
linuxstore.com.         1800    IN      NS      lap1.linuxstore.com.

;; ADDITIONAL SECTION:
lap2.linuxstore.com.    1800    IN      A       192.168.0.11

;; Query time: 5 msec
;; SERVER: 192.168.0.99#53(192.168.0.99)
;; WHEN: Thu Jan 23 19:58:32 2020
;; MSG SIZE  rcvd: 102
```

Agora, a partir do host 192.168.0.77 (ou seja, fora da lista dos autorizados), vamos também lançar o comando **dig**:

```
[root@curso8 ~]# dig @192.168.0.99 linuxstore.com

; <<>> DiG 9.8.1-P1 <<>> @192.168.0.99 lap1.linuxstore.com
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 62087
;; flags: qr aa rd; QUERY: 1, ANSWER: 1, AUTHORITY: 2, ADDITIONAL: 1
;; WARNING: recursion requested but not available

;; QUESTION SECTION:
;lap1.linuxstore.com.          IN    A

;; ANSWER SECTION:
lap1.linuxstore.com.    1800   IN    A     172.16.30.26

;; AUTHORITY SECTION:
linuxstore.com.         1800   IN    NS    lap2.linuxstore.com.
linuxstore.com.         1800   IN    NS    lap1.linuxstore.com.

;; ADDITIONAL SECTION:
lap2.linuxstore.com.    1800   IN    A     172.16.30.27

;; Query time: 19 msec
;; SERVER: 192.168.0.99#53(192.168.0.99)
;; WHEN: Thu Jan 23 19:56:54 2020
;; MSG SIZE  rcvd: 102
```

4.4 Execução do BIND em um Ambiente Chroot

Uma camada importante de segurança que poderá ser oferecida a diferentes serviços no Linux (inclusive o **DNS**) é a possibilidade do processo referente ser executado em um ambiente alterado. Neste sentido, entra em cena o **chroot**, que é uma operação que permite alterar o diretório raiz do sistema a um dado processo.

Ou seja, quando o **BIND**, por exemplo, é executado neste ambiente alterado, a raiz do sistema (para ele) não é mais o **/**, mas sim aquele diretório estabelecido pelo administrador – funcionando como uma espécie de jaula ao processo. Por isso, este ambiente alterado é conhecido como **chroot jail**.

Os próximos exemplos serão executados no host TESTE, cujo IP é o 192.168.0.77 e ainda não possui o **BIND** instalado. Desta forma, vamos instalar o pacote "bind-chroot":

```
[root@curso8 ~]# yum install bind-chroot -y
```

A partir de agora, o diretório **/var/named/chroot** é criado e, para este serviço, se torna então a sua raiz (e não mais o **/**). Observe o seu conteúdo:

```
[root@curso8 ~]# ls -l /var/named/chroot

total 0
drwxr-x--- 2 root named  6 Aug  8 09:16 dev
drwxr-x--- 4 root named 30 Jan 19 09:24 etc
drwxr-x--- 3 root named 19 Jan 19 09:24 run
drwxr-xr-x 3 root root  19 Jan 19 09:24 usr
drwxr-x--- 5 root named 52 Jan 19 09:24 var
```

Além disso, é importante destacar que, ao instalar o pacote "bind-chroot" (bem como o pacote "bind" apenas), é criado automaticamente um usuário e grupo para estarem associados ao processo do serviço.

Em distribuições baseadas em Red Hat, este usuário é identificado como "named". Ou seja, por razões de segurança, não é conveniente que o daemon do serviço seja executado através do usuário root.

Observe a pesquisa por tal usuário nos arquivos **/etc/passwd** e **/etc/group**:

```
[root@curso8 ~]# grep named /etc/passwd && grep named /etc/group

named:x:25:25:Named:/var/named/chroot:/sbin/nologin

named:x:25:
```

Neste momento, é importante parar o daemon **named** (caso esteja em execução) e, em seguida, desabilitá-lo:

```
[root@curso8 ~]# systemctl stop named

[root@curso8 ~]# systemctl disable named
```

Agora precisamos "ligar" o ambiente **chroot**:

```
[root@curso8 ~]# /usr/libexec/setup-named-chroot.sh /var/named/chroot on
```

O script de inicialização irá montar automaticamente todos os arquivos de configurações importantes ao **BIND** sob **/var/named/chroot**.

Agora, será preciso inicializar e habilitar o daemon **named-chroot**:

```
[root@curso8 ~]# systemctl start named-chroot

[root@curso8 ~]# systemctl enable named-chroot

[root@curso8 ~]# systemctl status named-chroot

● named-chroot.service - Berkeley Internet Name Domain (DNS)
   Loaded: loaded (/usr/lib/systemd/system/named-chroot.service; enabled; vendor
preset: disabled)
   Active: active (running) since Fri 2020-01-24 12:56:35 -03; 17min ago
 Main PID: 2476 (named)
   CGroup: /system.slice/named-chroot.service
           └─2476 /usr/sbin/named -u named -c /etc/named.conf -t
/var/named/chroot
```

Desta forma, veja que o daemon foi devidamente montado sob a jaula, ou seja, **/var/named/chroot**.

4.5 Uso Prático do TSIG

O Transaction Signatures (**TSIG**) é um recurso utilizado para assinar mensagens de **DNS** através do uso de chaves compartilhadas.

Disponível desde a versão 8.2 do **BIND**, este mecanismo costuma ser bastante utilizado para proteger as Transferências entre Zonas que são realizadas entre **nameservers**.

Para visualizarmos isto na prática, vamos trabalhar novamente com o cenário composto pelos hosts LAP1 e LAP2.

Estando no diretório **/var/named** do primeiro, lançamos o comando **dnssec-keygen** para gerar as chaves criptográficas. Além de gerar chaves **TSIG**, este mesmo comando também é utilizado para gerar chaves **DNSSEC**, como veremos mais a frente.

Desta forma, para gerar as chaves, lançamos:

```
[root@curso8 named]# dnssec-keygen -a HMAC-MD5 -b 512 -r /dev/urandom -n HOST
dlteclpic2.com

Kdlteclpic2.com.+157+41846
```

Observe um resumo das opções que foram usadas:

-a: permite que seja especificado o algoritmo a ser aplicado na geração das chaves. Neste exemplo, utilizamos o HMAC-MD5.

-b: permite que seja especificado o tamanho (em bits) da chave. Neste exemplo, estamos gerando chaves de 512 bits.

-r: permite especificar a origem que deverá ser utilizada para a geração de caracteres aleatórios. Neste exemplo, o dispositivo especial de caractere **/dev/urandom** foi utilizado.

-n: permite que seja especificado o tipo do dono da chave. Neste caso, utilizamos o tipo "HOST".

Por fim, especificamos um nome para estar associado às chaves. Neste momento, podemos listar o conteúdo do diretório corrente e perceber que as duas chaves foram criadas com sucesso – uma pública e outra privada:

```
[root@curso8 named]# ls -l K*

-rw------- 1 root root 123 Jan 20 08:37 Kdlteclpic2.com.+157+41846.key
-rw------- 1 root root 229 Jan 20 08:37 Kdlteclpic2.com.+157+41846.private
```

Vamos observar o conteúdo presente na chave privada:

```
[root@curso8 named]# cat Kestudoslpic2.com.br.+157+33565.private

Private-key-format: v1.3
Algorithm: 157 (HMAC_MD5)
Key:
ipzeRHLsMifUaIys1gaPLLMWnyyBX2edHhmKbfMxdgm2+yR2gMzJpobyCSU/K6nLSvJJ9svngQArDYh5U
GH37A==
Bits: AAA=
Created: 20200120113756
Publish: 20200120113756
Activate: 20200120113756
```

Para os nossos propósitos, o que interessa são as strings de caracteres contidas no campo Key. Sendo assim, vamos criar o arquivo **/var/named/dlteclpic2.com.key** e incluir as seguintes instruções:

```
key "dlteclpic2" {
algorithm HMAC-MD5;
      secret
"ipzeRHLsMifUaIys1gaPLLMWnyyBX2edHhmKbfMxdgm2+yR2gMzJpobyCSU/K6nLSvJJ9svngQArDYh5
UGH37A==";
};
```

Em seguida, vamos editar o arquivo **named.conf**, de forma a posicionar uma instrução **include** que será responsável por carregar o conteúdo do arquivo recém-criado:

```
include "/var/named/dlteclpic2.com.key";
```

Para que este recurso seja aplicado à zona "dlteclpic2.com", de forma que a Transferência entre Zonas seja realizada com o uso do **TSIG**, vamos até a definição da zona e a alteramos da seguinte forma:

```
zone "dlteclpic2.com" IN {
type master;
      file "dlteclpic2.com";
      notify yes;
      allow-update { none; };
      allow-transfer { key "dlteclpic2"; };
};
```

Para finalizar o processo no LAP1, após lançarmos o comando **named-checkconf** sobre o arquivo para verificar a sua sintaxe, vamos utilizar o comando **rndc** junto ao subcomando **reload**.

Estando agora no LAP2, vamos criar o arquivo **/etc/bind/named.conf.key**, onde iremos incluir a mesma definição de chave que realizamos no LAP1. Além disso, utilizaremos também a instrução **server** da seguinte maneira:

```
key "dlteclpic2" {
      algorithm HMAC-MD5;
      secret
"ipzeRHLsMifUaIys1gaPLLMWnyyBX2edHhmKbfMxdgm2+yR2gMzJpobyCSU/K6nLSvJJ9svngQArDYh5
UGH37A==";
};

server 192.168.0.15 {
      keys { dlteclpic2; };
};
```

Já no arquivo **/etc/bind/named.conf** utilizaremos o **include** para carregar o arquivo anterior:

```
include "/etc/bind/named.conf.key";
```

Outro arquivo que precisará ser editado é o **/etc/bind/named.conf.local**. Nele, a definição da zona deverá estar da seguinte maneira:

```
zone "dlteclpic2.com" IN {
      type slave;
      masters { 192.168.0.15; };
      file "dlteclpic2.com";
      allow-transfer { key "dlteclpic2"; };
};
```

Por fim, recarregamos os arquivos do daemon através do comando **rndc** junto ao subcomando **reload**.

Se verificarmos agora no arquivo **/var/log/syslog** do LAP2, notaremos que o processo de Transferência entre Zonas se deu de forma segura, a partir do uso do **TSIG**:

```
Jan 20 07:48:22 lap2 named[1948]: received control channel command 'retransfer
dlteclpic2.com'
Jan 20 07:48:22 lap2 named[1948]: zone dlteclpic2.com/IN: Transfer started.
Jan 20 07:48:22 lap2 named[1948]: transfer of 'dlteclpic2.com/IN' from
192.168.0.15#53: connected using 192.168.0.16#42827
Jan 20 07:48:22 lap2 named[1948]: zone dlteclpic2.com/IN: transferred serial 1:
TSIG 'dlteclpic2'
Jan 20 07:48:22 lap2 named[1948]: transfer of 'dlteclpic2.com/IN' from
192.168.0.15#53: Transfer status: success
Jan 20 07:48:22 lap2 named[1948]: transfer of 'dlteclpic2.com/IN' from
192.168.0.15#53: Transfer completed: 1 messages, 11 records, 369 bytes, 0.001
secs (369000 bytes/sec)
Jan 20 07:48:22 lap2 named[1948]: zone dlteclpic2.com/IN: sending notifies
(serial 1)
```

4.6 Noções Básicas sobre DNSSEC e DANE

As extensões **DNSSEC** (DNS Security Extensions), descritas nos **RFCs** 4033, 4034 e 4035, permitem a um cliente de **DNS** verificar a integridade e autenticidade das respostas oferecidas pelos **nameservers**. Com isso, pode-se então garantir que elas não sofreram qualquer alteração enquanto estavam em trânsito.

As consultas de **DNS** são essencialmente inseguras e, devido a esta característica, são sujeitas a interceptação de usuários mal intencionados. Os **nameservers** que atuam com consultas recursivas também não conseguem ter certeza que os registros obtidos a partir de outros nameservers sejam verdadeiros. Desta forma, o **DNSSEC** surgiu para atender essas e outras necessidades relacionadas a segurança.

Como já podemos imaginar, trata-se de um assunto bastante complexo e com uma extensa bibliografia. Porém, o **LPI** apenas espera que você possua uma noção geral sobre os aspectos mais importantes, além de alguns comandos que são utilizados neste contexto. Porém, é válido o aprofundamento posterior. Como o **DNS** é um serviço muito crítico, quanto mais informações você possuir, mais seguro e estável será o ambiente a ser administrado.

Para visualizarmos essas questões na prática e tendo sempre o foco naquilo que poderá ser cobrado no exame, vamos utilizar uma máquina com o hostname "sixlap1" (IP 192.168.0.27), rodando o CentOS e com o **BIND** instalado.

A primeira coisa a ser feita é verificar se o **DNSSEC** encontra-se habilitado no **nameserver** local. Para isto, as instruções **dnssec-enable** e **dnssec-validation** deverão estar configuradas como "yes" no arquivo **/etc/named.conf**:

```
dnssec-enable yes;
dnssec-validation yes;
```

Com o **DNSSEC**, é possível que você assine digitalmente uma zona usando uma chave criptográfica. Observe uma zona definida neste mesmo arquivo:

```
zone "casablanca.net" IN {
type master;
      file "casablanca.net";
      allow-update { none; };
};
```

Suas configurações específicas podem ser visualizadas no arquivo **/var/named/casablanca.net**:

```
$TTL 46800

@       IN      SOA     sixlap1.casablanca.net. contato.casablanca.net. (
1       ; -- serial
1D      ; -- refresh
1H      ; -- retry
1W      ; -- expire
3H      ; -- negative caching
)

                IN      NS      sixlap1
sixlap1         IN      A       192.168.0.27

                IN      NS      sixlap2
sixlap2         IN      A       192.168.0.28

                IN      MX      0       mail1
mail1           IN      A               192.168.0.29

                IN      MX      10      mail2
mail2           IN      A               192.168.0.30

www     IN      CNAME   sixlap1
```

Desta forma, vamos agora iniciar o processo conhecido como **ZSK** (Zone Signing Key). Ou seja, esse processo resultará na criação de duas chaves criptográficas (uma pública e outra privada). Essas são usadas para assinar digitalmente os registros de recursos referentes a esta zona. Estando sob **/var/named**, lançamos o comando **dnssec-keygen** da seguinte forma:

```
[root@curso8 named]# dnssec-keygen -a RSASHA256 -b 2048 -n ZONE casablanca.net.
```

Veja que agora utilizamos a opção **–n** junto à string **ZONE** para especificar o tipo do dono da chave. Em seguida, passamos o nome da zona, seguida pelo ponto final.

No final deste procedimento, as duas chaves são geradas em **/var/named**:

```
-rw-r--r-- 1 root  root   612 Jan 24 23:47 Kcasablanca.net.+008+25485.key
-rw------- 1 root  root  1776 Jan 24 23:47 Kcasablanca.net.+008+25485.private
```

Agora precisamos precisamos gerar outro par de chaves, responsável por assinar a **ZSK** referente à zona que estamos trabalhando. Este processo é conhecido como **KSK** (Key Signing Key). Para isto, também utilizaremos o comando **dnssec-keygen**, mas utilizaremos a opção **–f** para destacar a flag "KSK":

```
[root@curso8 named]# dnssec-keygen -f KSK -a RSASHA256 -b 2048 -n ZONE
casablanca.net.
```

Observe as novas chaves geradas no diretório **/var/named**:

```
-rw-r--r-- 1 root  root   611 Jan 24 23:48 Kcasablanca.net.+008+17350.key
-rw------- 1 root  root  1776 Jan 24 23:48 Kcasablanca.net.+008+17350.private
```

Em seguida, vamos abrir o arquivo das configurações específicas da zona e posicionar duas instruções **include**, contendo a localização das chaves públicas referentes a ambos os processos:

```
$include /var/named/Kcasablanca.net.+008+17350.key
$include /var/named/Kcasablanca.net.+008+25485.key
```

Por fim, vamos assinar a zona com as chaves públicas geradas pelos processos anteriores. Neste caso, utilizaremos o comando **dnssec-signzone**:

```
[root@curso8 named]# dnssec-signzone -x -o casablanca.net -K /var/named -k
Kcasablanca.net.+008+17350 casablanca.net Kcasablanca.net.+008+25485

Verifying the zone using the following algorithms: RSASHA256.
Zone fully signed:
Algorithm: RSASHA256: KSKs: 1 active, 0 stand-by, 0 revoked
                      ZSKs: 1 active, 0 present, 0 revoked
casablanca.net.signed
```

Observe um resumo das opções utilizadas:

-x: Assina a zona com o mínimo necessário.

-o: Especifica o nome da zona a ser trabalhada.

-K: Permite que seja especificado o diretório de localização das chaves.

-k: Especifica o nome da chave **KSK** ("Kcasablanca.net.+008+17350").

Por fim, são especificados o nome do arquivo referente à zona e o nome da chave **ZSK** ("Kcasablanca.net.+008+25485").

Nesse momento, o arquivo " casablanca.net.signed" é gerado em **/var/named**. Vamos até o arquivo **/etc/named.conf** e, nas definições da zona, especificamos este arquivo na instrução **file**:

```
zone "casablanca.net" {
        type master;
        file "casablanca.net.signed";
        allow-update { none; };
};
```

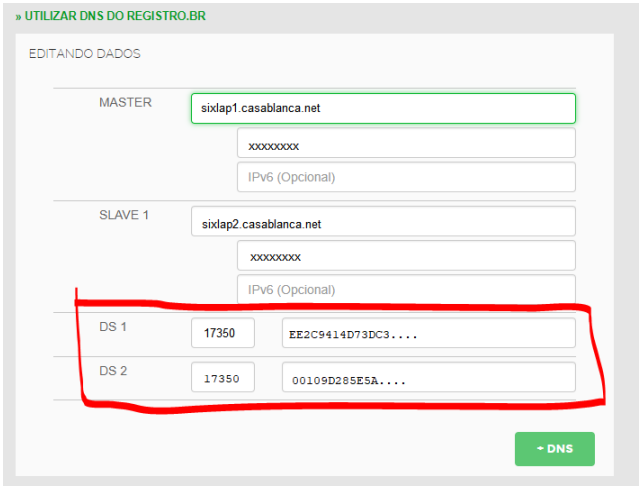
Para finalizar, lançamos o comando **rndc** junto ao subcomando **reload**.

Quando lançamos o comando **dnssec-signzone**, outro arquivo importante também foi gerado, denominado "dsset-casablanca.net." Observe o seu conteúdo:

```
[root@curso8 ~]#  cat /var/named/dsset-casablanca.net.

casablanca.net.         IN DS 17350 8 1 EE2C9414D73DC319C9AE5287F4CF0BD2E4594FAA
casablanca.net.         IN DS 17350 8 2
00109D285E5A5308EBA936017C217A53A9DB9F547D11E7A708861223 B3719B80
```

Esses dados poderão ser utilizados no momento de registro de um domínio. Esses registros **DS** devem então ser concedidos à zona pai através de seus portais. Observe um exemplo:



Para visualizarmos a chave pública da zona que criamos, alteramos a instrução **recursion** para "no". Em seguida, a partir de outro host, lançamos o comando **dig** junto à opção **DNSKEY**:

```
[root@curso8 ~]#  dig @192.168.0.27 DNSKEY casablanca.net +multiline

; <<>> DiG 9.11.4-P2-RedHat-9.11.4-9.P2.el7 <<>> @192.168.0.27 DNSKEY
casablanca.net +multiline
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 6880
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 2, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
```

```
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;casablanca.net.                IN DNSKEY

;; ANSWER SECTION:
casablanca.net.         46800 IN DNSKEY 256 3 8 (
                               AwEAAdm8tv/5whHCy3r/ncPxkJzy8Gtalm48GO0Yewwo
                               nUIivkureCdnEThVd8wTjTcVFAPQfppmILUxmi4caXQm
                               hJrjWUdJSl9P3pf51/W7RrrqvajnkNqF6b6jJJUdh3O0
                               uubgNPywQ3bnPUZLwU/ZtRIwTu7KWQ2soeFeeWWxsp1q
                               mmeM/iPi9yRsuK/HOqm0jBbXR7lio8BQeH8Ndv+lydex
                               hAcRdiNsQvGLrsCpfW9Jg8atxKgBQna8Ao4TysxbcEOU
                               ZRWpGbg7fX7j3h//i8hCes/rIFnzQ1b/lQfU+Z5wIH6H
                               Jh7zIeiNQWFmM4nOp/IhwBcLHX5MEeAVh77FKbs=
                               ) ; ZSK; alg = RSASHA256 ; key id = 25485
casablanca.net.         46800 IN DNSKEY 257 3 8 (
                               AwEAAdrVELtkO1vlnMq1ALmWe9AzTnkuPQWke8eAJ8Ll
                               gs4hzRP37wVOYVA3FvNYsf8r+si7cLTaURqYnzbjUaEr
                               k1rdv26qfCoU0+ZK4gTQiAOTrUfRTMPa0HPmFDLqaplA
                               0pvpVbI38wpUz5Uu7GJRMl6WDWzLHU6s+Omb7owdknfG
                               vMwXvxYAH/7Y2vlvIqFKSC3ibVLzhgoTfGkfQl2UTyIT
                               ZzR2QWtBmODK0HPcNH+C9nx5/UgjBoYKOvfWs6htPFvj
                               p8fWbVbUZnN5jNZ3fn60FtnrPdhAccUJB9x2yuoP23KZ
                               wg8oR1+NQ7W+quJi8A+hbrQ0VCs0N1f0uwAEwU8=
                               ) ; KSK; alg = RSASHA256 ; key id = 17350

;; Query time: 0 msec
;; SERVER: 192.168.0.27#53(192.168.0.27)
;; WHEN: Sat Jan 25 00:24:49 -03 2020
;; MSG SIZE  rcvd: 595
```

Para visualizar os registros **RRSIG**, lançamos:

```
[root@curso8 ~]#  dig @192.168.0.27 casablanca.net +dnssec +multiline

; <<>> DiG 9.11.4-P2-RedHat-9.11.4-9.P2.el7 <<>> @192.168.0.27 casablanca.net
+dnssec +multiline
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 55386
;; flags: qr aa rd; QUERY: 1, ANSWER: 0, AUTHORITY: 4, ADDITIONAL: 1
;; WARNING: recursion requested but not available

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags: do; udp: 4096
;; QUESTION SECTION:
;casablanca.net.                IN A

;; AUTHORITY SECTION:
casablanca.net.         10800 IN SOA sixlap1.casablanca.net.
contato.casablanca.net. (
                               1          ; serial
                               86400      ; refresh (1 day)
                               3600       ; retry (1 hour)
                               604800     ; expire (1 week)
                               10800      ; minimum (3 hours)
                               )
casablanca.net.         10800 IN RRSIG SOA 8 2 46800 (
```

```
                                20200224015300 20200125015300 25485
casablanca.net.
                                blj1wmX7o7Z0H2/tVvV6bVeAMRUMwqCIT1HlWzwsyaXO
                                udQrsYL3uVau6e2fWLcrWfLUI8ZoGB+RusswxwSSLGBD
                                hodGgwiZkeqVHUZ0lc6UxaMkrTh5Ppyvxg3Hn2MEDhyr
                                327wRIdUjj/aEw76QQbSWsiJxlLFVgOSNuYZK/SYnD4/
                                5uxJWoxq4iZ22/KMv6u7l/jJHPB3Zfs/WhlmffwhHUl6
                                Clid3S7KBUclx2Ii8LS4r1W8vkqwRCisU9VVh2LgN2h2
                                ZjlRGHDzstdjsNsFFLu6y1L2f8fmlGUrPTAGahBOvZ3D
                                boq2q8o4b3PjjE2SdjrIMw2yusjtIkVBtA== )
casablanca.net.         10800 IN NSEC mail1.casablanca.net. NS SOA RRSIG NSEC
DNSKEY
casablanca.net.         10800 IN RRSIG NSEC 8 2 10800 (
                                20200224015300 20200125015300 25485
casablanca.net.
                                FQtTbNskzFHITmgBl79Mk1c+QVnjiOkIGHInREluhVGs
                                sl98gMRDIZw4np8QMODJFYJ+x8WEqI8jmxQXeK3NAWlP
                                WwICnLRPkYj3V6COqKUrFRrSb4sFso8kNZVeyD8pECLj
                                +plbNVLGUuf+3dFSKvae87wk8P6Karol4cvGBvWlZvY7
                                /gZNpOcy64PwqbHlhQeodxHm2EWSXbUXnFuR88R/jatS
                                Cg9rda1SW61eIZ68VvVTn08v64aAdII2cseEYyhfeCk8
                                HkfF9l7laHb3pK7wilKbQjCN70gQC0XEiVAko0cCu1Oh
                                q3BfJ8ZFzTxfxjW9mGuGeHRfxdVxcloLhA== )

;; Query time: 0 msec
;; SERVER: 192.168.0.27#53(192.168.0.27)
;; WHEN: Sat Jan 25 00:34:45 -03 2020
;; MSG SIZE  rcvd: 742
```

Note que cada resposta possui um assinatura digital correspondente (**RRSIG**). Para visualizarmos outra questão, vamos até "sixlap1" e alterar novamente a instrução **recursion** para "yes".

Em seguida, a partir dele próprio, lançamos o comando **dig** rumo ao domínio "whitehouse.gov":

```
[root@curso8 ~]#  dig @localhost whitehouse.gov +dnssec +multiline

; <<>> DiG 9.11.4-P2-RedHat-9.11.4-9.P2.el7 <<>> @localhost whitehouse.gov
+dnssec +multiline
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 24761
;; flags: qr rd ra ad; QUERY: 1, ANSWER: 2, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags: do; udp: 4096
;; QUESTION SECTION:
;whitehouse.gov.                IN A

;; ANSWER SECTION:
whitehouse.gov.         20 IN A 23.197.12.199
whitehouse.gov.         20 IN RRSIG A 7 2 20 (
                                20200128033302 20200125023302 47918
whitehouse.gov.
                                BGUDewXipoDLE1xBu7I9WWxTbM9+PQfnHY734d4VbBfX
                                xyUuaHoC48DkKViYXYzaMgWvE1Ue+nrb+XSyHMq3oL45
                                YCwC9uAeDBPsUzBLIZkfFnhNJSgqNce+zk4UDyPI9ZzV
                                NDtFYw8XT7RsImf8jxeBjRcfi797r8XAVrinlAI= )
```

```
;; Query time: 3701 msec
;; SERVER: ::1#53(::1)
;; WHEN: Sat Jan 25 00:41:56 -03 2020
;; MSG SIZE  rcvd: 233
```

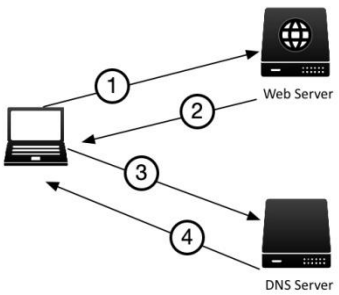
Uma resposta validada deverá possuir a flag **AD** (Authenticated Data) configurado e o cabeçalho não deverá possuir o status **NOERROR**.

Após configurar o **DNSSEC** para um dado domínio, o administrador também poderá utilizar o **DANE** (DNS-based Authentication of Named Entities): protocolo de segurança que permite o vínculo de certificados digitais **X.509** (geralmente usados via **TLS** – Transport Layer Security) a nomes de domínio que estejam usando o **DNSSEC**.

Quando o protocolo **TLS** é empregado na comunicação entre cliente e um servidor web, por exemplo, parte do processo envolve o uso de um certificado gerado e assinado por terceiros – conhecidos como **CA** (Certificate Authority). Porém, além de existirem centenas de **CAs**, qualquer um deles consegue gerar certificados digitais para qualquer domínio. Em conseqüência disso, nos últimos anos vários deles sofreram problemas relacionados à segurança, permitindo a geração de certificados de domínios para aqueles que não eram os seus donos.

Um navegador web, por exemplo, ao aceitar um certificado comprometido, pode permitir que uma chave (também comprometida) seja instalada. Dessa forma, um atacante pode usar esta chave para descriptografar e ler o tráfego **TCP**.

O **DANE** entra em cena nesse cenário permitindo que o administrador de um domínio ateste as chaves usadas nos clientes ou servidores **TLS** de tal domínio. Ou seja, ele associa o certificado ou chave pública oficial e correta do servidor web (ou da autoridade confiável que esteja emitindo o certificado) ao nome de domínio correto usando uma consulta de **DNS**. Observe a lógica:



1 – O navegador cliente se conecta ao endereço https://www.exemple.com.

2 – O servidor web informa o certificado.

3 – O cliente consulta o seu servidor de **DNS** local pelo registro **TLSA** do www.exemple.com.

4 – O servidor de **DNS** realiza uma pesquisa comum pelo registro **TLSA** do www.exemple.com. Além disso, utiliza o **DNSSEC** para validar a resposta obtida através dos **nameservers** autoritativos deste domínio.

Após receber o registro **TLSA** validado, o navegador registra esta informação e o compara com o certificado recebido a partir do servidor web. Caso os dois não forem iguais, o navegador exibe um aviso e não carrega a página.

Observe um exemplo de registro **TLSA** para o domínio "www.exemple.com":

```
_443._tcp.www.example.com. IN TLSA 3 1 1 8bd1da95272f7fa4ffb24137fc0ed03aae67e5c4d8b3c50734e1050a7920b922
```

Neste exemplo, os administradores de "exemple.com" pegaram o certificado que publicaram para o website, geraram um hash SHA-256 e publicaram este hash como um registro de recurso **TLSA** no **DNS**. Este procedimento poderá ser feito usando ferramentas open source, como o OpenSSL.

5 Conclusão e Certificado

5.1 Conclusão e Certificado

Parabéns por ter chegado ao final do curso **SERVIDOR DE NOMES DE DOMÍNIO NO LINUX**!

Tenha certeza de que compreendeu todos os conceitos aqui mostrados.

Não esqueça que você deve ler e assistir tudo para obter o seu certificado de conclusão do curso.

Ficamos por aqui e nos encontramos nos próximos cursos!!!!!