



Red Hat Enterprise Linux 8

管理网络基础架构服务

在 Red Hat Enterprise Linux 8 中管理网络基础架构服务的指南

Red Hat Enterprise Linux 8 管理网络基础架构服务

在 Red Hat Enterprise Linux 8 中管理网络基础架构服务的指南

法律通告

Copyright © 2024 Red Hat, Inc.

The text of and illustrations in this document are licensed by Red Hat under a Creative Commons Attribution–Share Alike 3.0 Unported license ("CC-BY-SA"). An explanation of CC-BY-SA is available at

<http://creativecommons.org/licenses/by-sa/3.0/>

. In accordance with CC-BY-SA, if you distribute this document or an adaptation of it, you must provide the URL for the original version.

Red Hat, as the licensor of this document, waives the right to enforce, and agrees not to assert, Section 4d of CC-BY-SA to the fullest extent permitted by applicable law.

Red Hat, Red Hat Enterprise Linux, the Shadowman logo, the Red Hat logo, JBoss, OpenShift, Fedora, the Infinity logo, and RHCE are trademarks of Red Hat, Inc., registered in the United States and other countries.

Linux[®] is the registered trademark of Linus Torvalds in the United States and other countries.

Java[®] is a registered trademark of Oracle and/or its affiliates.

XFS[®] is a trademark of Silicon Graphics International Corp. or its subsidiaries in the United States and/or other countries.

MySQL[®] is a registered trademark of MySQL AB in the United States, the European Union and other countries.

Node.js[®] is an official trademark of Joyent. Red Hat is not formally related to or endorsed by the official Joyent Node.js open source or commercial project.

The OpenStack[®] Word Mark and OpenStack logo are either registered trademarks/service marks or trademarks/service marks of the OpenStack Foundation, in the United States and other countries and are used with the OpenStack Foundation's permission. We are not affiliated with, endorsed or sponsored by the OpenStack Foundation, or the OpenStack community.

All other trademarks are the property of their respective owners.

摘要

本文档论述了如何在 Red Hat Enterprise Linux 8 中设置和管理网络核心基础架构服务，如 DNS 和 DHCP。

目录

对红帽文档提供反馈	3
第 1 章 设置和配置 BIND DNS 服务器	4
1.1. 有关使用 SELINUX 保护 BIND 的注意事项，或者在更改 ROOT 环境中运行	4
1.2. BIND 管理员参考手册	4
1.3. 将 BIND 配置为缓存 DNS 服务器	5
1.4. 在 BIND DNS 服务器中配置日志记录	7
1.5. 编写 BIND ACL	8
1.6. 在 BIND DNS 服务器中配置区	10
1.7. 在 BIND DNS 服务器中配置区传输	18
1.8. 在 BIND 中配置响应策略区以覆盖 DNS 记录	21
1.9. 使用 DNSTAP 记录 DNS 查询	24
第 2 章 设置 UNBOUND DNS 服务器	26
2.1. 将 UNBOUND 配置为缓存 DNS 服务器	26
第 3 章 提供 DHCP 服务	28
3.1. 静态和动态 IP 地址之间的区别	28
3.2. DHCP 的阶段	28
3.3. 对 DHCPV4 和 DHCPV6 使用 DHCPD 时的不同	28
3.4. DHCPD 服务的租期数据库	29
3.5. DHCPV6 和 RADVD 的比较	29
3.6. 为 IPV6 路由器配置 RADVD 服务	30
3.7. 为 DHCP 服务器设置网络接口	31
3.8. 为直接连接到 DHCP 服务器的子网设置 DHCP 服务	32
3.9. 为没有直接连接到 DHCP 服务器的子网设置 DHCP 服务	35
3.10. 使用 DHCP 为主机分配静态地址	38
3.11. 使用 GROUP 声明同时将参数应用到多个主机、子网和共享网络	39
3.12. 恢复损坏的租期数据库	41
3.13. 设置 DHCP 转发代理	42

对红帽文档提供反馈

我们感谢您对我们文档的反馈。让我们了解如何改进它。

通过 Jira 提交反馈（需要帐户）

1. 登录到 [Jira](#) 网站。
2. 单击顶部导航栏中的 **Create**。
3. 在 **Summary** 字段中输入描述性标题。
4. 在 **Description** 字段中输入您对改进的建议。包括到文档相关部分的链接。
5. 点对话框底部的 **Create**。

第 1 章 设置和配置 BIND DNS 服务器

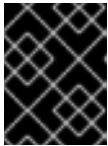
BIND 是一个功能丰富的 DNS 服务器，它完全符合互联网工程任务 Force (IETF) DNS 标准和草案标准。例如，管理员经常使用 BIND，如下所示：

- 在本地网络中缓存 DNS 服务器
- 区域的权威 DNS 服务器
- 二级服务器来为区域提供高可用性

1.1. 有关使用 SELINUX 保护 BIND 的注意事项，或者在更改 ROOT 环境中运行

要保护 BIND 安装，您可以：

- 运行 **named** 而不需要 **change-root** 环境。在这种情况下，**enforcing** 模式中的 SELinux 会阻止利用已知的 BIND 安全漏洞。默认情况下，Red Hat Enterprise Linux 在 **enforcing** 模式中使用 SELinux。



重要

在 SELinux 处于 **enforcing** 模式的 RHEL 上运行 BIND 比在 **change-root** 环境中运行 BIND 更安全。

- 在 **change-root** 环境中运行 **named-chroot** 服务。
利用 **change-root** 功能，管理员可以定义进程的根目录及其子进程与 / 目录不同。当您启动 **named-chroot** 服务时，BIND 将其根目录切换到 **/var/named/chroot/**。因此，服务使用 **mount --bind** 命令使 **/etc/named-chroot.files** 中列出的文件和目录保存在 **/var/named/chroot/** 中，并且进程无法访问 **/var/named/chroot/** 以外的文件。

如果您决定使用 BIND：

- 在正常模式中，使用 **named** 服务。
- 在 **change-root** 环境中，使用 **named-chroot** 服务。这要求您安装 **named-chroot** 软件包。

其他资源

- **named(8)** man page 的 **Red Hat SELinux BIND security profile** 部分

1.2. BIND 管理员参考手册

bind 软件包中包括的综合 **BIND 管理员参考手册**：

- 配置示例
- 有关高级功能的文档
- 配置参考
- 安全考虑

要在安装了 **bind** 软件包的主机上显示 **BIND Administrator Reference Manual**，请在浏览器中打开 `/usr/share/doc/bind/Bv9ARM.html` 文件。

1.3. 将 BIND 配置为缓存 DNS 服务器

默认情况下，BIND DNS 服务器解析和缓存成功并失败的查找。随后，服务会从其缓存中应答相同记录的请求。这可显著提高 DNS 查找速度。

前提条件

- 服务器的 IP 地址是静态的。

流程

1. 安装 **bind** 和 **bind-utils** 软件包：

```
# yum install bind bind-utils
```

这些软件包提供 BIND 9.11。如果您需要 BIND 9.16，请安装 **bind9.16** 和 **bind9.16-utils** 软件包。

2. 如果要在 `chroot` 环境中运行 BIND，请安装 **bind-chroot** 软件包：

```
# yum install bind-chroot
```

请注意，在 SELinux 处于 **enforcing** 模式（默认设置）的主机上运行 BIND 更为安全。

3. 编辑 `/etc/named.conf` 文件，并在 **options** 语句中进行以下更改：

- a. 更新 **listen-on** 和 **listen-on-v6** 语句，以指定 BIND 应该侦听的 IPv4 和 IPv6 接口：

```
listen-on port 53 { 127.0.0.1; 192.0.2.1; };  
listen-on-v6 port 53 { ::1; 2001:db8:1::1; };
```

- b. 更新 **allow-query** 语句，以配置哪些 IP 地址和范围客户端可以查询此 DNS 服务器：

```
allow-query { localhost; 192.0.2.0/24; 2001:db8:1::/64; };
```

- c. 添加 **allow-recursion** 语句，以定义 BIND 接受递归查询的 IP 地址和范围：

```
allow-recursion { localhost; 192.0.2.0/24; 2001:db8:1::/64; };
```



警告

不要在服务器的公共 IP 地址中递归。否则，服务器可能会成为大规模 DNS 扩大攻击的一部分。

- d. 默认情况下，BIND 通过将从根服务器递归查询到权威 DNS 服务器来解析查询。或者，您可以将 BIND 配置为将查询转发到其他 DNS 服务器，比如您的供应商之一。在这种情况下，添加一个带有 BIND 应该转发查询的 DNS 服务器的 IP 地址列表的 **forwarders** 语句：

```
forwarders { 198.51.100.1; 203.0.113.5; };
```

作为回退行为，如果转发器服务器没有响应，BIND 会以递归方式解析查询。要禁用此行为，请添加 **forward only;** 语句。

4. 验证 `/etc/named.conf` 文件的语法：

```
# named-checkconf
```

如果命令没有显示输出，则语法为正确的。

5. 更新 **firewalld** 规则，以允许传入的 DNS 流量：

```
# firewall-cmd --permanent --add-service=dns  
# firewall-cmd --reload
```

6. 启动并启用 BIND：

```
# systemctl enable --now named
```

如果要在 `change-root` 环境中运行 BIND，请使用 **systemctl enable --now named-chroot** 命令启用并启动该服务。

验证

1. 使用新设置 DNS 服务器解析域：

```
# dig @localhost www.example.org  
...  
www.example.org. 86400 IN A 198.51.100.34  
  
;; Query time: 917 msec  
...
```

本例假定 BIND 在同一主机上运行并响应 **localhost** 接口上的查询。

在第一次查询记录后，BIND 会将条目添加到其缓存中。

2. 重复前面的查询：

```
# dig @localhost www.example.org  
...  
www.example.org. 85332 IN A 198.51.100.34  
  
;; Query time: 1 msec  
...
```

由于对条目进行了缓存，进一步对相同记录的请求会非常快，直到条目过期为止。

后续步骤

- 配置网络中的客户端来使用此 DNS 服务器。如果 DHCP 服务器向客户端提供 DNS 服务器设置，请相应地更新 DHCP 服务器的配置。

其他资源

- [有关使用 SELinux 保护 BIND 的注意事项，或者在更改 root 环境中运行](#)
- [named.conf\(5\) man page](#)
- [/usr/share/doc/bind/sample/etc/named.conf](#)
- [BIND 管理员参考手册](#)

1.4. 在 BIND DNS 服务器中配置日志记录

默认 `/etc/named.conf` 文件中的配置（如 `bind` 软件包提供）使用 `default_debug` 通道，并将消息记录到 `/var/named/data/named.run` 文件中。`default_debug` 频道仅在服务器的 debug 级别为零时记录条目。

使用不同的频道和类别，您可以将 BIND 配置为将具有定义的严重性的不同事件写入单独的文件。

前提条件

- 已配置了 BIND，例如作为缓存名称服务器。
- `named` 或 `named-chroot` 服务正在运行。

流程

1. 编辑 `/etc/named.conf` 文件，并将 `category` 和 `channel` 添加到 `logging` 语句中，例如：

```
logging {
    ...

    category notify { zone_transfer_log; };
    category xfer-in { zone_transfer_log; };
    category xfer-out { zone_transfer_log; };
    channel zone_transfer_log {
        file "/var/named/log/transfer.log" versions 10 size 50m;
        print-time yes;
        print-category yes;
        print-severity yes;
        severity info;
    };

    ...
};
```

使用这个示例配置，BIND 会记录与区域传送相关的消息，到 `/var/named/log/transfer.log`。BIND 创建最多 **10** 个日志文件版本，如果它们达到 **50 MB** 的最大大小，则轮转它们。

`category` 定义了 BIND 向哪些频道发送类别信息。

`channel` 定义了日志消息的目的地，包括版本数量、最大文件大小以及 BIND 应记录到频道的严重性等级。其他设置（如启用日志的时间戳、类别和严重性）是可选的，但可用于调试目的。

2. 如果不存在，创建日志目录，并为 **named** 用户授予对这个目录的写权限：

```
# mkdir /var/named/log/
# chown named:named /var/named/log/
# chmod 700 /var/named/log/
```

3. 验证 **/etc/named.conf** 文件的语法：

```
# named-checkconf
```

如果命令没有显示输出，则语法为正确的。

4. 重启 BIND：

```
# systemctl restart named
```

如果在 **change-root** 环境中运行 BIND，请使用 **systemctl restart named-chroot** 命令来重启该服务。

验证

- 显示日志文件的内容：

```
# cat /var/named/log/transfer.log
...
06-Jul-2022 15:08:51.261 xfer-out: info: client @0x7fecbc0b0700 192.0.2.2#36121/key
example-transfer-key (example.com): transfer of 'example.com/IN': AXFR started: TSIG
example-transfer-key (serial 2022070603)
06-Jul-2022 15:08:51.261 xfer-out: info: client @0x7fecbc0b0700 192.0.2.2#36121/key
example-transfer-key (example.com): transfer of 'example.com/IN': AXFR ended
```

其他资源

- **named.conf(5)** man page
- [BIND 管理员参考手册](#)

1.5. 编写 BIND ACL

控制 BIND 的某些功能的访问可以防止未经授权的访问和攻击，如拒绝服务 (DoS)。BIND 访问控制列表 (**acl**) 语句是 IP 地址和范围的列表。每个 ACL 都有一个别名，您可以在几个语句中使用，如 **allow-query** 来引用指定的 IP 地址和范围。



警告

BIND 仅在 ACL 中使用第一个匹配条目。例如，如果您定义了 ACL { **192.0.2/24;** **!192.0.2.1;** } 以及带有 **192.0.2.1** IP 地址的主机的连接，即使第二个条目排除这个地址，也会授予访问权限。

BIND 有以下内置 ACL :

- **none** : 不匹配主机。
- **any**: 匹配所有主机。
- **localhost** : 匹配回环地址 **127.0.0.1** 和 **::1**, 以及服务器上运行 BIND 的服务器上的所有接口的 IP 地址。
- **localnets** : 匹配回环地址 **127.0.0.1** 和 **::1**, 以及运行 BIND 的服务器都直接连接到的所有子网。

前提条件

- 已配置了 BIND, 例如作为缓存名称服务器。
- **named** 或 **named-chroot** 服务正在运行。

流程

1. 编辑 **/etc/named.conf** 文件并进行以下更改 :

- a. 将 **acl** 语句添加到文件中。例如, 要为 **127.0.0.1**、**192.0.2.0/24** 和 **2001:db8:1::/64** 创建名为 **internal-networks** 的 ACL, 请输入 :

```
acl internal-networks { 127.0.0.1; 192.0.2.0/24; 2001:db8:1::/64; };
acl dmz-networks { 198.51.100.0/24; 2001:db8:2::/64; };
```

- b. 在支持它们的声明中使用 ACL 的别名, 例如 :

```
allow-query { internal-networks; dmz-networks; };
allow-recursion { internal-networks; };
```

2. 验证 **/etc/named.conf** 文件的语法 :

```
# named-checkconf
```

如果命令没有显示输出, 则语法为正确的。

3. 重新载入 BIND :

```
# systemctl reload named
```

如果在 **change-root** 环境中运行 BIND, 请使用 **systemctl reload named-chroot** 命令来重新加载该服务。

验证

- 执行操作, 以触发使用配置的 ACL 的功能。例如, 此流程中的 ACL 只允许来自定义的 IP 地址的递归查询。在这种情况下, 在不属于 ACL 定义的主机上输入以下命令来尝试解析外部域 :

```
# dig +short @192.0.2.1 www.example.com
```

如果命令没有返回任何输出，BIND 拒绝访问，且 ACL 可以正常工作。有关客户端的详细输出，请使用不带 **+short** 选项的命令：

```
# dig @192.0.2.1 www.example.com
...
;; WARNING: recursion requested but not available
...
```

其他资源

- [BIND Administrator Reference Manual](#) 中的 **Access control lists** 部分。

1.6. 在 BIND DNS 服务器中配置区

DNS 区域是包含域空间中特定子树的资源记录的数据库。例如，如果您负责 **example.com** 域，可以在 BIND 中为它设置一个区。因此，客户端可将 **www.example.com** 解析为在这个区中配置的 IP 地址。

1.6.1. 区域文件中的 SOA 记录

SOA (start of authority) 记录在一个 DNS 区中是必必需的记录。例如，如果多个 DNS 服务器对某个区域具有权威，那么此记录非常重要，但也指向 DNS 解析器。

BIND 中的 SOA 记录具有以下语法：

```
name class type mname rname serial refresh retry expire minimum
```

为提高可读性，管理员通常将区域文件中的记录分成多行，其中包含以分号(;)开头的注释。请注意，如果您分割 SOA 记录，圆括号将记录保留在一起：

```
@ IN SOA ns1.example.com. hostmaster.example.com. (
    2022070601 ; serial number
    1d      ; refresh period
    3h      ; retry period
    3d      ; expire time
    3h )    ; minimum TTL
```

重要

请注意完全限定域名 (FQDN) 末尾的结尾点。FQDN 包含多个域标签，它们用点分开。由于 DNS root 有一个空标签，所以 FQDN 以点结尾。因此，BIND 在没有结尾点的情况下将区域名称附加到名称中。不含尾部点的主机名（例如 **ns1.example.com**）会被扩展为 **ns1.example.com.example.com.**，对于主域名服务器，这不是正确的地址。

以下是 SOA 记录中的字段：

- **name**：区域的名称，即所谓的 **源 (origin)**。如果将此字段设置为 **@**，BIND 会将其扩展为 **/etc/named.conf** 中定义的区域名称。
- **class**：在 SOA 记录中，必须将此字段始终设置为 Internet (**IN**)。
- **type**：在 SOA 记录中，必须将此字段始终设置为 **SOA**。
- **mname**（主名称）：此区域的主域名服务器的主机名。

- **rname** (负责名称): 负责此区域的电子邮件地址。请注意, 格式不同。您必须将 at 符号 (@) 替换为点 (.)。
- **serial** : 此区域文件的版本号。次要域名服务器仅在主服务器上的序列号较高时更新其区域副本。
格式可以是任意数字值。通常的格式是 **<year><month><day><two-digit-number>**。使用这种格式, 在理论上可以每天修改区域文件上百次。
- **refresh** : 在检查主服务器更新时, 次要服务器应等待的时间。
- **retry** : 当次要服务器在尝试失败后重试查询主服务器的时间长度。
- **expire** : 当所有之前的尝试失败时, 次要服务器停止查询主服务器的时间长度。
- **minimum** : RFC 2308 将此字段的含义改为负缓存时间。兼容解析器使用它来确定缓存 **NXDOMAIN** 名称错误的时间。



注意

refresh, retry, expire, 和 minimum 项中的值定义了一个时间 (以秒为单位)。但是, 为了提高可读性, 请使用时间后缀, 如 **m** 表示分钟、**h** 表示小时, 以及 **d** 表示天。例如, **3h** 代表 3 小时。

其他资源

- [RFC 1035](#): 域名 - 实现和规格
- [RFC 1034](#) : 域名 - 概念和功能
- [RFC 2308](#) : DNS 查询 (DNS 缓存) 的 Negative 缓存

1.6.2. 在 BIND 主服务器上设置转发区

转发区域将名称映射到 IP 地址和其他信息。例如, 如果您负责域 **example.com**, 您可以在 BIND 中设置转发区来解析名称, 如 **www.example.com**。

前提条件

- 已配置了 BIND, 例如作为缓存名称服务器。
- **named** 或 **named-chroot** 服务正在运行。

流程

1. 在 **/etc/named.conf** 文件中添加区定义 :

```
zone "example.com" {
    type master;
    file "example.com.zone";
    allow-query { any; };
    allow-transfer { none; };
};
```

这些设置定义 :

- 此服务器作为 **example.com** 区域的主服务器 (类型 **master**)。
- **/var/named/example.com.zone** 文件是区域文件。如果您设置了相对路径，如本例中所示，这个路径相对于您在 **options** 语句中的目录中创建的 **directory** 相对。
- 任何主机都可以查询此区域。另外，还可指定 IP 范围或 BIND 访问控制列表 (ACL) 别名来限制访问。
- 没有主机可以传输区域。仅在设置次要服务器并且仅为次要服务器的 IP 地址时才允许区域传送。

2. 验证 **/etc/named.conf** 文件的语法：

```
# named-checkconf
```

如果命令没有显示输出，则语法为正确的。

3. 使用以下内容创建 **/var/named/example.com.zone** 文件：

```
$TTL 8h
@ IN SOA ns1.example.com. hostmaster.example.com. (
    2022070601 ; serial number
    1d      ; refresh period
    3h      ; retry period
    3d      ; expire time
    3h )    ; minimum TTL

    IN NS  ns1.example.com.
    IN MX  10 mail.example.com.

www      IN A   192.0.2.30
www      IN AAAA 2001:db8:1::30
ns1      IN A   192.0.2.1
ns1      IN AAAA 2001:db8:1::1
mail     IN A   192.0.2.20
mail     IN AAAA 2001:db8:1::20
```

这个区域文件：

- 将资源记录的默认生存时间 (TTL) 值设置为 8 小时。如果没有时间后缀（例如没有使用 **h** 指定小时），BIND 会将该值解析为秒。
- 包含所需的 SOA 资源记录，以及有关该区域的详细信息。
- 将 **ns1.example.com** 设置为此区域的权威 DNS 服务器。要正常工作，区域需要至少一个域名服务器 (NS) 记录。但是，若要与 RFC 1912 兼容，您需要至少有两个域名服务器。
- 将 **mail.example.com** 设置为 **example.com** 域的邮件交换器 (MX)。主机名前面的数字值是记录的优先级。较低值的条目具有更高的优先级。
- 设置 **www.example.com** 的 IPv4 和 IPv6 地址、**mail.example.com** 和 **ns1.example.com**。

4. 在区域文件上设置安全权限，仅允许 **named** 组读取它：


```
# chown root:named /var/named/example.com.zone
# chmod 640 /var/named/example.com.zone
```

5. 验证 `/var/named/example.com.zone` 文件的语法：

```
# named-checkzone example.com /var/named/example.com.zone
zone example.com/IN: loaded serial 2022070601
OK
```

6. 重新载入 BIND：

```
# systemctl reload named
```

如果在 `change-root` 环境中运行 BIND，请使用 `systemctl reload named-chroot` 命令来重新加载该服务。

验证

- 从 `example.com` 区域查询不同的记录，并验证输出是否与您在区域文件中配置的记录匹配：

```
# dig +short @localhost AAAA www.example.com
2001:db8:1::30

# dig +short @localhost NS example.com
ns1.example.com.

# dig +short @localhost A ns1.example.com
192.0.2.1
```

本例假定 BIND 在同一主机上运行并响应 `localhost` 接口上的查询。

其他资源

- [区域文件中的 SOA 记录](#)
- [编写 BIND ACL](#)
- [BIND 管理员参考手册](#)
- [RFC 1912 - 通用 DNS 操作和配置错误](#)

1.6.3. 在 BIND 主服务器中设置反向区

反向区域将 IP 地址映射到名称。例如，如果您负责 IP 范围 `192.0.2.0/24`，您可以在 BIND 中设置反向区域，以将 IP 地址从这个范围内的 IP 地址解析为主机名。



注意

如果您为整个类网络创建一个反向区域，请相应地命名区域。例如，对于 C network `192.0.2.0/24`，区域的名称是 `2.0.192.in-addr.arpa`。如果要为不同的网络大小创建反向区域，如 `192.0.2.0/28`，区域的名称为 `28-2.0.192.in-addr.arpa`。

前提条件

- 已配置了 BIND，例如作为缓存名称服务器。
- `named` 或 `named-chroot` 服务正在运行。

流程

1. 在 `/etc/named.conf` 文件中添加区定义：

```
zone "2.0.192.in-addr.arpa" {
    type master;
    file "2.0.192.in-addr.arpa.zone";
    allow-query { any; };
    allow-transfer { none; };
};
```

这些设置定义：

- 此服务器作为 `2.0.192.in-addr.arpa` 反向区域的主服务器(`type master`)。
 - `/var/named/2.0.192.in-addr.arpa.zone` 文件是区域文件。如果您设置了相对路径，如本例中所示，这个路径相对于您在 `options` 语句中的目录中创建的 `directory` 相对。
 - 任何主机都可以查询此区域。另外，还可指定 IP 范围或 BIND 访问控制列表 (ACL) 别名来限制访问。
 - 没有主机可以传输区域。仅在设置次要服务器并且仅为次要服务器的 IP 地址时才允许区域传送。
2. 验证 `/etc/named.conf` 文件的语法：

```
# named-checkconf
```

如果命令没有显示输出，则语法为正确的。

3. 使用以下内容创建 `/var/named/2.0.192.in-addr.arpa.zone` 文件：

```
$TTL 8h
@ IN SOA ns1.example.com. hostmaster.example.com. (
    2022070601 ; serial number
    1d      ; refresh period
    3h      ; retry period
    3d      ; expire time
    3h )    ; minimum TTL

    IN NS  ns1.example.com.

1      IN PTR ns1.example.com.
30     IN PTR www.example.com.
```

这个区域文件：

- 将资源记录的默认生存时间 (TTL) 值设置为 8 小时。如果没有时间后缀（例如没有使用 `h` 指定小时），BIND 会将该值解析为秒。
- 包含所需的 SOA 资源记录，以及有关该区域的详细信息。

- 将 **ns1.example.com** 设置为此反向区域的权威 DNS 服务器。要正常工作，区域需要至少一个域名服务器 (NS) 记录。但是，若要与 RFC 1912 兼容，您需要至少有两个域名服务器。
 - 设置 **192.0.2.1** 和 **192.0.2.30** 地址的指针(PTR)记录。
4. 在区域文件上设置安全权限，仅允许 **named** 组读取它：

```
# chown root:named /var/named/2.0.192.in-addr.arpa.zone
# chmod 640 /var/named/2.0.192.in-addr.arpa.zone
```

5. 验证 **/var/named/2.0.192.in-addr.arpa.zone** 文件的语法：

```
# named-checkzone 2.0.192.in-addr.arpa /var/named/2.0.192.in-addr.arpa.zone
zone 2.0.192.in-addr.arpa/IN: loaded serial 2022070601
OK
```

6. 重新载入 BIND：

```
# systemctl reload named
```

如果在 **change-root** 环境中运行 BIND，请使用 **systemctl reload named-chroot** 命令来重新加载该服务。

验证

- 从反向区查询不同的记录，并验证输出是否与您配置在区域文件中的记录匹配：

```
# dig +short @localhost -x 192.0.2.1
ns1.example.com.

# dig +short @localhost -x 192.0.2.30
www.example.com.
```

本例假定 BIND 在同一主机上运行并响应 **localhost** 接口上的查询。

其他资源

- [区域文件中的 SOA 记录](#)
- [编写 BIND ACL](#)
- [BIND 管理员参考手册](#)
- [RFC 1912 - 通用 DNS 操作和配置错误](#)

1.6.4. 更新 BIND 区文件

在某些情况下，例如，如果服务器的 IP 地址有变化，您必须更新区域文件。如果多个 DNS 服务器负责某个区，则仅在主服务器中执行这个步骤。存储区域副本的其他 DNS 服务器将通过区域传送接收更新。

前提条件

- zone 被配置。

- **named** 或 **named-chroot** 服务正在运行。

流程

1. 可选：识别 `/etc/named.conf` 文件中的区文件的路径：

```
options {
    ...
    directory    "/var/named";
}

zone "example.com" {
    ...
    file "example.com.zone";
};
```

您可以在区域定义的 **file** 指令中找到到区域文件的路径。相对路径相对于 **options** 语句中的 **directory** 设置的相对路径。

2. 编辑区域文件：
 - a. 进行必要的更改。
 - b. 在 SOA 记录中递增序列号。



重要

如果序列号等于或低于先前值，次要服务器不会更新其区域的副本。

3. 验证区文件的语法：

```
# named-checkzone example.com /var/named/example.com.zone
zone example.com/IN: loaded serial 2022062802
OK
```

4. 重新载入 BIND：

```
# systemctl reload named
```

如果在 `change-root` 环境中运行 BIND，请使用 **systemctl reload named-chroot** 命令来重新加载该服务。

验证

- 查询您添加、修改或删除的记录，例如：

```
# dig +short @localhost A ns2.example.com
192.0.2.2
```

本例假定 BIND 在同一主机上运行并响应 **localhost** 接口上的查询。

其他资源

- [区域文件中的 SOA 记录](#)

- 在 BIND 主服务器上设置转发区
- 在 BIND 主服务器中设置反向区

1.6.5. 使用自动密钥生成和区维护功能进行 DNSSEC 区域签名

您可以使用域名系统安全扩展 (DNSSEC) 为区域签名，以确保身份验证和数据完整性。此类区域包含额外的资源记录。客户端可以使用它们来验证区域信息的真实性。

如果您为区启用 DNSSEC 策略功能，BIND 会自动执行以下操作：

- 创建密钥
- 为区域签名
- 维护区域，包括重新签名并定期替换密钥。



重要

要启用外部 DNS 服务器以验证区的真实性，您必须在父区中添加该区域的公钥。请联系您的域供应商或 registry，以了解更多有关如何完成此操作的详细信息。

此流程使用 BIND 中的内置 **default** DNSSEC 策略。这个策略使用单一 **ECDSAP256SHA** 密钥签名。另外，还可创建自己的策略来使用自定义密钥、算法和计时。

前提条件

- 已安装 BIND 9.16 或更高版本。要满足此要求，请安装 **bind9.16** 软件包，而不是 **bind**。
- 配置您要启用 DNSSEC 的区域。
- **named** 或 **named-chroot** 服务正在运行。
- 服务器可将时间与时间服务器同步。对于 DNSSEC 验证，系统时间准确非常重要。

流程

1. 编辑 **/etc/named.conf** 文件，并将 **dnssec-policy default**；添加到您要启用 DNSSEC 的区域：

```
zone "example.com" {
    ...
    dnssec-policy default;
};
```

2. 重新载入 BIND：

```
# systemctl reload named
```

如果在 **change-root** 环境中运行 BIND，请使用 **systemctl reload named-chroot** 命令来重新加载该服务。

3. BIND 将公钥存储在 **/var/named/K<zone_name>.<algorithm>.<key_ID>.key** 文件中。使用此文件显示区的公钥，格式为父区所需的格式：

- DS 记录格式：

```
# dnssec-dsfromkey /var/named/Kexample.com.+013+61141.key
example.com. IN DS 61141 13 2
3E184188CF6D2521EDFDC3F07CFEE8D0195AACBD85E68BAE0620F638B4B1B027
```

- DNSKEY 格式：

```
# grep DNSKEY /var/named/Kexample.com.+013+61141.key
example.com. 3600 IN DNSKEY 257 3 13
sjzT3jNEp120aSO4mPEHHSkReHUf7AABNnT8hNRTzD5cKMQSjDJin2l3
5CaKVcWO1pm+HltxUEt+X9dfp8OZkg==
```

4. 请求将区域的公钥添加到父区。请联系您的域供应商或 registry，以了解更多有关如何完成此操作的详细信息。

验证

1. 从启用了 DNSSEC 签名的区域查询您自己的 DNS 服务器：

```
# dig +dnssec +short @localhost A www.example.com
192.0.2.30
A 13 3 28800 20220718081258 20220705120353 61141 example.com.
e7Cfh6GuOBMAWsgsHSVTPH+JJSOI/Y6zctzluqlU1JqEgOOAfL/Qz474
M0sgj54m1Kmnr2ANBKJN9uvOs5eXYw==
```

本例假定 BIND 在同一主机上运行并响应 **localhost** 接口上的查询。

2. 在将公钥添加到父区并传播到其他服务器后，验证服务器是否将查询上的身份验证数据(**ad**)标记设置为已签名区域：

```
# dig @localhost example.com +dnssec
...
;; flags: qr rd ra ad; QUERY: 1, ANSWER: 2, AUTHORITY: 0, ADDITIONAL: 1
...
```

其他资源

- [在 BIND 主服务器上设置转发区](#)
- [在 BIND 主服务器中设置反向区](#)

1.7. 在 BIND DNS 服务器中配置区传输

区域传送可确保所有具有区域副本的 DNS 服务器均使用最新数据。

前提条件

- 在未来的主服务器中，已配置要设置区域传送的区域。
- 在未来的次要服务器上，已配置 BIND，例如作为缓存名称服务器。
- 在两个服务器上，**named** 或 **named-chroot** 服务正在运行。

流程

1. 在现有主服务器中：

- a. 创建一个共享密钥，并将其附加到
- `/etc/named.conf`
- 文件中：

```
# tsig-keygen example-transfer-key | tee -a /etc/named.conf
key "example-transfer-key" {
    algorithm hmac-sha256;
    secret "q7ANbnyliDMuvWgnKOxMLi313JGcTZB5ydMW5CyUGXQ=";
};
```

这个命令显示 `tsig-keygen` 命令的输出，并自动将其附加到 `/etc/named.conf` 中。

稍后，在次要服务器上，您还需要命令的输出。

- b. 编辑
- `/etc/named.conf`
- 文件中的区定义：

- i. 在
- `allow-transfer`
- 语句中，定义服务器必须提供
- `example-transfer-key`
- 语句中指定的密钥来传输区：

```
zone "example.com" {
    ...
    allow-transfer { key example-transfer-key; };
};
```

另外，在 `allow-transfer` 语句中使用 BIND 访问控制列表 (ACL) 别名。

- ii. 默认情况下，在更新区域后，BIND 会通知所有在区中有名称服务器 (NS) 记录的域名服务器。如果您不计划为二级服务器添加 NS 记录，您可以配置 BIND 通知这个服务器。为此，请将这个次要服务器的 IP 地址添加
- `also-notify`
- 声明到区：

```
zone "example.com" {
    ...
    also-notify { 192.0.2.2; 2001:db8:1::2; };
};
```

- c. 验证
- `/etc/named.conf`
- 文件的语法：

```
# named-checkconf
```

如果命令没有显示输出，则语法为正确的。

- d. 重新载入 BIND：

```
# systemctl reload named
```

如果在 `change-root` 环境中运行 BIND，请使用 `systemctl reload named-chroot` 命令来重新加载该服务。

2. 在未来的次要服务器中：

- a. 按如下方式编辑
- `/etc/named.conf`
- 文件：

- i. 添加与主服务器相同的密钥定义：

```
key "example-transfer-key" {
    algorithm hmac-sha256;
```

```
secret "q7ANbnyliDMuvWgnKOxMLi313JGcTZB5ydMW5CyUGXQ=";
};
```

- ii. 在 `/etc/named.conf` 文件中添加区定义：

```
zone "example.com" {
    type slave;
    file "slaves/example.com.zone";
    allow-query { any; };
    allow-transfer { none; };
    masters {
        192.0.2.1 key example-transfer-key;
        2001:db8:1::1 key example-transfer-key;
    };
};
```

这些设置状态：

- 此服务器是 **example.com** 区域的次要服务器 (**type slave**)。
- `/var/named/slaves/example.com.zone` 文件是区域文件。如果您设置了相对路径，如本例中所示，这个路径相对于您在 **options** 语句中的目录中创建的 **directory** 相对。要隔离此服务器从属的区域文件，您可以将它们存储在 `/var/named/slaves/` 目录中。
- 任何主机都可以查询此区域。另外，还可指定 IP 范围或 ACL 别名来限制访问。
- 没有主机可以从该服务器传输区域。
- 此区域的主服务器的 IP 地址是 **192.0.2.1** 和 **2001:db8:1::2**。或者，您可以指定 ACL 别名。此次要服务器将使用名为 **example-transfer-key** 的键向主服务器进行身份验证。

- b. 验证 `/etc/named.conf` 文件的语法：

```
# named-checkconf
```

- c. 重新载入 BIND：

```
# systemctl reload named
```

如果在 `change-root` 环境中运行 BIND，请使用 `systemctl reload named-chroot` 命令来重新加载该服务。

3. 可选：修改主服务器上的区域文件，并为新的次要服务器添加一个 **NS** 记录。

验证

在次要服务器中：

1. 显示 **named** 服务的 **systemd** 日志条目：

```
# journalctl -u named
...
Jul 06 15:08:51 ns2.example.com named[2024]: zone example.com/IN: Transfer started.
Jul 06 15:08:51 ns2.example.com named[2024]: transfer of 'example.com/IN' from
```



```
192.0.2.1#53: connected using 192.0.2.2#45803
Jul 06 15:08:51 ns2.example.com named[2024]: zone example.com/IN: transferred serial
2022070101
Jul 06 15:08:51 ns2.example.com named[2024]: transfer of 'example.com/IN' from
192.0.2.1#53: Transfer status: success
Jul 06 15:08:51 ns2.example.com named[2024]: transfer of 'example.com/IN' from
192.0.2.1#53: Transfer completed: 1 messages, 29 records, 2002 bytes, 0.003 secs (667333
bytes/sec)
```

如果在 `change-root` 环境中运行 BIND，请使用 `journalctl -u named-chroot` 命令显示日志条目。

2. 验证 BIND 创建了区域文件：

```
# ls -l /var/named/slaves/
total 4
-rw-r--r--. 1 named named 2736 Jul 6 15:08 example.com.zone
```

请注意，默认情况下，次要服务器以二进制原始格式存储区域文件。

3. 从次要服务器查询传输的区的记录：

```
# dig +short @192.0.2.2 AAAA www.example.com
2001:db8:1::30
```

本例假定您在此流程中设置的次要服务器侦听 IP 地址 **192.0.2.2**。

其他资源

- [在 BIND 主服务器上设置转发区](#)
- [在 BIND 主服务器中设置反向区](#)
- [编写 BIND ACL](#)
- [更新 BIND 区文件](#)

1.8. 在 BIND 中配置响应策略区以覆盖 DNS 记录

使用 DNS 块和过滤，管理员可以重写 DNS 响应来阻止对某些域或主机的访问。在 BIND 中，响应策略区域 (RPZ) 提供此功能。您可以为受阻条目配置不同的操作，如返回 **NXDOMAIN** 错误或不响应查询。

如果您的环境中有多多个 DNS 服务器，请使用此流程在主服务器上配置 RPZ，稍后配置区传输以在您的次要服务器上提供 RPZ。

前提条件

- 已配置了 BIND，例如作为缓存名称服务器。
- `named` 或 `named-chroot` 服务正在运行。

流程

1. 编辑 `/etc/named.conf` 文件并进行以下更改：

- a. 在 **options** 语句中添加 **response-policy** 定义：

```
options {
    ...

    response-policy {
        zone "rpz.local";
    };

    ...
}
```

您可以在 **response-policy** 的 **zone** 语句中为 RPZ 设置自定义名称。但是，在下一步中，您必须在区定义中使用相同的名称。

- b. 为您在上一步中设置的 RPZ 添加 **zone** 定义：

```
zone "rpz.local" {
    type master;
    file "rpz.local";
    allow-query { localhost; 192.0.2.0/24; 2001:db8:1::/64; };
    allow-transfer { none; };
};
```

这些设置状态：

- 此服务器是名为 **rpz.local** 的 RPZ 的主服务器 (**type master**)。
- **/var/named/rpz.local** 文件是区域文件。如果您设置了相对路径，如本例中所示，这个路径相对于您在 **options** 语句中的目录中创建的 **directory** 相对。
- **allow-query** 中定义的任何主机都可以查询此 RPZ。另外，还可指定 IP 范围或 BIND 访问控制列表 (ACL) 别名来限制访问。
- 没有主机可以传输区域。仅在设置次要服务器并且仅为次要服务器的 IP 地址时才允许区域传送。

2. 验证 **/etc/named.conf** 文件的语法：

```
# named-checkconf
```

如果命令没有显示输出，则语法为正确的。

3. 使用以下内容创建 **/var/named/rpz.local** 文件，例如：

```
$TTL 10m
@ IN SOA ns1.example.com. hostmaster.example.com. (
    2022070601 ; serial number
    1h       ; refresh period
    1m       ; retry period
    3d       ; expire time
    1m )     ; minimum TTL

    IN NS  ns1.example.com.

example.org  IN CNAME .
```

```
*.example.org  IN CNAME .
example.net   IN CNAME rpz-drop.
*.example.net IN CNAME rpz-drop.
```

这个区域文件：

- 将资源记录的默认生存时间 (TTL) 值设置为 10 分钟。如果没有时间后缀（例如没有使用 **h** 指定小时），BIND 会将该值解析为秒。
- 包含所需的 SOA 资源记录，以及有关该区域的详细信息。
- 将 **ns1.example.com** 设置为此区域的权威 DNS 服务器。要正常工作，区域需要至少一个域名服务器 (NS) 记录。但是，若要与 RFC 1912 兼容，您需要至少有两个域名服务器。
- 将查询的 **NXDOMAIN** 错误返回给该域中的 **example.org** 和主机。
- 将查询丢弃至此域中的 **example.net** 和主机。

有关操作和示例的完整列表，请参阅 [IETF 草案：DNS 响应策略区域\(RPZ\)](#)。

4. 验证 `/var/named/rpz.local` 文件的语法：

```
# named-checkzone rpz.local /var/named/rpz.local
zone rpz.local/IN: loaded serial 2022070601
OK
```

5. 重新载入 BIND：

```
# systemctl reload named
```

如果在 `change-root` 环境中运行 BIND，请使用 `systemctl reload named-chroot` 命令来重新加载该服务。

验证

1. 尝试解析 **example.org** 中的主机，该主机在 RPZ 中配置，以返回 **NXDOMAIN** 错误：

```
# dig @localhost www.example.org
...
;; ->HEADER<<- opcode: QUERY, status: NXDOMAIN, id: 30286
...
```

本例假定 BIND 在同一主机上运行并响应 `localhost` 接口上的查询。

2. 尝试解析 **example.net** 域中的主机，该域在 RPZ 中配置以丢弃查询：

```
# dig @localhost www.example.net
...
;; connection timed out; no servers could be reached
...
```

其他资源

- [IETF 草案：DNS 响应策略区域\(RPZ\)](#)

1.9. 使用 DNSTAP 记录 DNS 查询

作为网络管理员，您可以记录域名系统(DNS)详情来分析 DNS 流量模式、监控 DNS 服务器性能，并对 DNS 问题进行故障排除。如果您希望有一个高级方法来监控和记录传入名称查询的详细信息，请使用 **dnstap** 接口记录从 **named** 服务发送的消息。您可以捕获并记录 DNS 查询，来收集有关网站或 IP 地址的信息。

前提条件

- **bind-9.11.26-2** 软件包或更新的版本已安装。



警告

如果您已安装并运行了 **BIND** 版本，添加新版本的 **BIND** 将覆盖现有的版本。

流程

1. 通过编辑 **options** 块中的 **/etc/named.conf** 文件来启用 **dnstap** 和目标文件：

```
options
{
# ...
dnstap { all; }; # Configure filter
dnstap-output file "/var/named/data/dnstap.bin";
# ...
};
# end of options
```

2. 要指定您要记录的 DNS 流量类型，请将 **dnstap** 过滤器添加到 **/etc/named.conf** 文件中的 **dnstap** 块中。您可以使用以下过滤器：

- **auth** - 权威区域响应或回答。
- **client** - 内部客户端查询或回答。
- **forwarder** - 转发的查询或来自它的响应。
- **resolver** - 迭代的解析查询或响应。
- **update** - 动态区域更新请求。
- **all** - 以上选项中的任何一个。
- **query** 或 **response** - 如果您没有指定 **query** 或 **response** 关键字，则 **dnstap** 两个都记录。



注意

dnstap 过滤器包含多个由 ; 分隔的定义，**dnstap {}** 块的语法如下：**dnstap{(all | auth | client | forwarder | resolver | update)[(query | response)]; ... };**

3. 要应用您的更改，请重启 **named** 服务：

```
# systemctl restart named.service
```

4. 为活跃日志配置定期回滚

在以下示例中，**cron** 调度程序每天运行一次用户编辑的脚本的内容。值为 **3** 的 **roll** 选项指定 **dnstap** 最多可以创建三个备份日志文件。值 **3** 覆盖 **dnstap-output** 变量的 **version** 参数，并将备份日志文件数限制为三个。此外，二进制日志文件被移到另一个目录并被重命名，并且永远不会达到 **.2** 后缀，即使三个备份文件已存在。如果根据大小限制自动回滚二进制日志足够了，则您可以跳过这一步。

Example:

```
sudoedit /etc/cron.daily/dnstap
```

```
#!/bin/sh
```

```
rndc dnstap -roll 3
```

```
mv /var/named/data/dnstap.bin.1 /var/log/named/dnstap/dnstap-$(date -l).bin
```

```
# use dnstap-read to analyze saved logs
```

```
sudo chmod a+x /etc/cron.daily/dnstap
```

5. 使用 **dnstap-read** 工具以人类可读的格式处理和分析日志：
在以下示例中，**dnstap-read** 工具以 **YAML** 文件格式打印输出。

Example:

```
dnstap-read -y [file-name]
```

第 2 章 设置 UNBOUND DNS 服务器

unbound DNS 服务器是一个验证、递归并进行缓存的 DNS 解析器。此外，**unbound** 侧重于安全性，例如，它会默认启用域名系统安全扩展 (DNSSEC)。

2.1. 将 UNBOUND 配置为缓存 DNS 服务器

默认情况下，**unbound** DNS 服务解析并缓存成功和失败的查找。随后，服务会从其缓存中应答相同记录请求。

流程

1. 安装 **unbound** 软件包：

```
# yum install unbound
```

2. 编辑 `/etc/unbound/unbound.conf` 文件，并在 **server** 子句中进行以下更改：

- a. 添加 **interface** 参数来配置 **unbound** 服务侦听查询的 IP 地址，例如：

```
interface: 127.0.0.1
interface: 192.0.2.1
interface: 2001:db8:1::1
```

使用这些设置时，**unbound** 仅侦听指定的 IPv4 和 IPv6 地址。

将接口限制到特定接口，可防止客户端从未经授权的网络（如互联网）向这个 DNS 服务器发送查询。

- b. 添加 **access-control** 参数，以配置客户端可以从哪些子网查询 DNS 服务，例如：

```
access-control: 127.0.0.0/8 allow
access-control: 192.0.2.0/24 allow
access-control: 2001:db8:1::/64 allow
```

3. 创建用于远程管理 **unbound** 服务的私钥和证书：

```
# systemctl restart unbound-keygen
```

如果您跳过这一步，在下一步中验证配置将会报告缺少的文件。如果文件丢失，**unbound** 服务自动创建这些文件。

4. 验证配置文件：

```
# unbound-checkconf
unbound-checkconf: no errors in /etc/unbound/unbound.conf
```

5. 更新 **firewalld** 规则，以允许传入的 DNS 流量：

```
# firewall-cmd --permanent --add-service=dns
# firewall-cmd --reload
```

6. 启用并启动 **unbound** 服务：

```
# systemctl enable --now unbound
```

验证

1. 查询在 **localhost** 接口上侦听的 **unbound** DNS 服务器以解析域：

```
# dig @localhost www.example.com
...
www.example.com. 86400 IN A 198.51.100.34

;; Query time: 330 msec
...
```

在第一次查询记录后，**unbound** 会将条目添加到其缓存中。

2. 重复前面的查询：

```
# dig @localhost www.example.com
...
www.example.com. 85332 IN A 198.51.100.34

;; Query time: 1 msec
...
```

由于对条目进行了缓存，进一步对相同记录的请求会非常快，直到条目过期为止。

后续步骤

- 配置网络中的客户端来使用此 DNS 服务器。例如，使用 **nmcli** 实用程序在 NetworkManager 连接配置集中设置 DNS 服务器的 IP：

```
# nmcli connection modify Example_Connection ipv4.dns 192.0.2.1
# nmcli connection modify Example_Connection ipv6.dns 2001:db8:1::1
```

其他资源

- **unbound.conf(5)** 手册页

第 3 章 提供 DHCP 服务

动态主机配置协议(DHCP)是一种网络协议，可以自动向客户端分配 IP 信息。您可以设置 **dhcpd** 服务，来在网络中提供 DHCP 服务器和 DHCP 转发。

3.1. 静态和动态 IP 地址之间的区别

静态 IP 地址

当您为某个设备分配静态 IP 地址时，该地址不会随时间变化，除非您手动更改该地址。如果您要使用静态 IP 地址：

- 确保 DNS 等服务器的网络地址一致性以及验证服务器。
- 使用独立于其他网络基础结构的带外管理设备。

动态 IP 地址

当您将设备配置为使用动态 IP 地址时，该地址会随时更改。因此，动态地址通常用于偶尔连接到网络的设备，因为重启主机后 IP 地址可能会不同。

动态 IP 地址更为灵活，更容易设置和管理。动态主机控制协议 (DHCP) 是动态为主机分配网络配置的传统方法。



注意

没有严格的规则来定义何时使用静态或动态 IP 地址。它取决于用户的需求、喜好和网络环境。

3.2. DHCP 的阶段

DHCP 分为四个阶段：Discovery、Offer、Request、Afirmldgement（也称为 DORA 进程）。DHCP 使用这个过程为客户端提供 IP 地址。

Discovery（发现）

DHCP 客户端发送一条信息来发现网络中 DHCP 服务器。这个消息在网络和数据链路层广播。

Offer（提供）

DHCP 服务器从客户端接收信息，并为 DHCP 客户端提供 IP 地址。这个消息在数据链路层单播，但在网络层广播。

Request（请求）

DHCP 客户端为提供的 IP 地址请求 DHCP 服务器。这个消息在数据链路层单播，但在网络层广播。

Acknowledgment（承认）

DHCP 服务器向 DHCP 客户端发送承认信息。这个消息在数据链路层单播，但在网络层广播。它是 DHCP DORA 进程的最后一个信息。

3.3. 对 DHCPV4 和 DHCPV6 使用 DHCPD 时的不同

dhcpd 服务支持在一个服务器上提供 DHCPv4 和 DHCPv6。但是，您需要单独的 **dhcpd** 实例以及单独的配置文件，来为每个协议提供 DHCP。

DHCPv4

- 配置文件：`/etc/dhcp/dhcpd.conf`

- systemd 服务名称：**dhcpcd**

DHCPv6

- 配置文件：**/etc/dhcp/dhcpd6.conf**
- systemd 服务名称：**dhcpd6**

3.4. DHCPD 服务的租期数据库

DHCP 租期是 **dhcpcd** 服务为客户端分配网络地址的期限。**dhcpcd** 服务将 DHCP 租期存储在以下数据库中：

- 对于 DHCPv4：**/var/lib/dhcpd/dhcpd.leases**
- 对于 DHCPv6：**/var/lib/dhcpd/dhcpd6.leases**



警告

手动更新数据库文件可能会破坏数据库。

租期数据库包含有关分配的租期的信息，如分配给 MAC 地址的 IP 地址或租期到期的时间戳。请注意，租期数据库中的所有时间戳都是 UTC。

dhcpcd 服务定期重新创建数据库：

1. 该服务重命名现有文件：
 - 将 **/var/lib/dhcpd/dhcpd.leases** 重命名为 **/var/lib/dhcpd/dhcpd.leases~**
 - **/var/lib/dhcpd/dhcpd6.leases** 重命名为 **/var/lib/dhcpd/dhcpd6.leases~**
2. 服务将所有已知的租期写入新创建的 **/var/lib/dhcpd/dhcpd.leases** 和 **/var/lib/dhcpd/dhcpd6.leases** 文件。

其他资源

- [dhcpcd.leases\(5\) 手册页](#)
- [恢复损坏的租期数据库](#)

3.5. DHCPV6 和 RADVD 的比较

在 IPv6 网络中，只有路由器广告消息提供有关 IPv6 默认网关的信息。因此，如果您需要在需要默认网关设置的子网中使用 DHCPv6，您必须额外配置路由器广告服务，如路由器广告守护进程(**radvd**)。

radvd 服务使用路由器广告数据包中的标志来声明 DHCPv6 服务器的可用性。

下表比较了 DHCPv6 和 **radvd** 的功能：

	DHCPv6	radvd
提供有关默认网关的信息	否	是
保证随机地址以保护隐私	是	否
发送更多网络配置选项	是	否
将 MAC 地址映射到 IPv6 地址	是	否

3.6. 为 IPV6 路由器配置 RADVD 服务

路由器广告守护进程(**radvd**)发送 IPv6 无状态自动配置所需的路由器广告消息。这可以让用户根据这些公告自动配置其地址、设置、路由和选择默认路由器。



注意

您只能在 **radvd** 服务中设置 /64 前缀。要使用其他前缀，请使用 DHCPv6。

先决条件

- 已以 **root** 用户身份登录。

步骤

1. 安装 **radvd** 软件包：

```
# yum install radvd
```

2. 编辑 **/etc/radvd.conf** 文件，并添加以下配置：

```
interface enp1s0
{
  AdvSendAdvert on;
  AdvManagedFlag on;
  AdvOtherConfigFlag on;

  prefix 2001:db8:0:1::/64 {
  };
};
```

这些设置将 **radvd** 配置为对 **2001:db8:0:1::/64** 子网在 **enp1s0** 设备上发送路由器广告消息。**AdvManagedFlag on** 设置定义了客户端应从 DHCP 服务器接收 IP 地址，而 **AdvOtherConfigFlag** 参数设为 **on**，定义了客户端也应从 DHCP 服务器接收非地址信息。

3. 另外，可以将 **radvd** 配置为在系统启动时自动启动：

```
# systemctl enable radvd
```

4. 启动 **radvd** 服务：

```
# systemctl start radvd
```

5. 另外，可以显示路由器广告数据包的内容，以及 **radvd** 发送的配置值：

```
# radvdump
```

其他资源

- **radvd.conf(5)** man page
- `/usr/share/doc/radvd/radvd.conf.example` file
- [我是否可以在 IPv6 路由器广告中使用不是 64 位的前缀长度？](#)

3.7. 为 DHCP 服务器设置网络接口

默认情况下，**dhcpd** 服务仅处理网络接口上的请求，该接口在在服务的配置文件有一个在子网中定义的 IP 地址。

例如，在以下场景中，**dhcpd** 仅侦听 **enp0s1** 网络接口：

- 在 `/etc/dhcp/dhcpd.conf` 文件中，您只有 `192.0.2.0/24` 的 **子网** 定义。
- **enp0s1** 网络接口连接到 `192.0.2.0/24` 子网。
- **enp7s0** 接口连接到不同的子网。

只有 DHCP 服务器包含连接到同一网络的多个网络接口，但服务应该只侦听特定的接口时，才应按照以下流程操作。

根据您要为 IPv4、IPv6 或两个协议提供 DHCP 的信息，请查看以下操作过程：

- [IPv4 网络](#)
- [IPv6 网络](#)

先决条件

- 已以 **root** 用户身份登录。
- **dhcp-server** 软件包已安装。

步骤

- 对于 IPv4 网络：
 1. 将 `/usr/lib/systemd/system/dhcpd.service` 文件复制到 `/etc/systemd/system/` 目录中：

```
# cp /usr/lib/systemd/system/dhcpd.service /etc/systemd/system/
```

不要编辑 `/usr/lib/systemd/system/dhcpd.service` 文件。**dhcp-server** 软件包的将来更新可以覆盖更改。

2. 编辑 `/etc/systemd/system/dhcpd.service` 文件，并追加接口名称，**dhcpd** 应该侦听 **ExecStart** 参数中的命令：

```
ExecStart=/usr/sbin/dhcpd -f -cf /etc/dhcp/dhcpd.conf -user dhcpd -group dhcpd --no-pid
$DHCPDARGS enp0s1 enp7s0
```

本例配置了 **dhcpd** 仅侦听 **enp0s1** 和 **enp7s0** 接口。

3. 重新加载 **systemd** 管理器配置：

```
# systemctl daemon-reload
```

4. 重启 **dhcpd** 服务：

```
# systemctl restart dhcpd.service
```

- 对于 IPv6 网络：

1. 将 **/usr/lib/systemd/system/dhcpd6.service** 文件复制到 **/etc/systemd/system/** 目录中：

```
# cp /usr/lib/systemd/system/dhcpd6.service /etc/systemd/system/
```

不要编辑 **/usr/lib/systemd/system/dhcpd6.service** 文件。**dhcp-server** 软件包的将来更新可以覆盖更改。

2. 编辑 **/etc/systemd/system/dhcpd6.service** 文件，并追加接口名称，**dhcpd** 应该侦听 **ExecStart** 参数中的命令：

```
ExecStart=/usr/sbin/dhcpd -f -6 -cf /etc/dhcp/dhcpd6.conf -user dhcpd -group dhcpd --no-pid
$DHCPDARGS enp0s1 enp7s0
```

本例配置了 **dhcpd** 仅侦听 **enp0s1** 和 **enp7s0** 接口。

3. 重新加载 **systemd** 管理器配置：

```
# systemctl daemon-reload
```

4. 重启 **dhcpd6** 服务：

```
# systemctl restart dhcpd6.service
```

3.8. 为直接连接到 DHCP 服务器的子网设置 DHCP 服务

如果 DHCP 服务器直接连接到该服务器应响应 DHCP 请求的子网，请使用以下步骤。如果服务器的网络接口有这个子网的 IP 地址，那么就会出现这种情况。

根据您要为 IPv4、IPv6 或两个协议提供 DHCP 的信息，请查看以下操作过程：

- [IPv4 网络](#)
- [IPv6 网络](#)

先决条件

- 已以 **root** 用户身份登录。

- **dhcp-server** 软件包已安装。

步骤

- 对于 IPv4 网络：

1. 编辑 `/etc/dhcp/dhcpd.conf` 文件：

- 另外，如果没有其他指令包含这些设置，则添加 **dhcpd** 用作默认值的全局参数：

```
option domain-name "example.com";
default-lease-time 86400;
```

本例将连接 `example.com` 的默认域名设为 **example.com**，默认的租期时间设为 **86400** 秒（1 天）。

- 在新行中添加 **authoritative** 语句：

```
authoritative;
```



重要

没有 **authoritative** 语句，如果客户端询问池外部的地址，则 **dhcpd** 服务不会回答带有 **DHCPNAK** 的 **DHCPREQUEST** 消息。

- 对于每个直接连接到服务器接口的 IPv4 子网，请添加 **subnet** 声明：

```
subnet 192.0.2.0 netmask 255.255.255.0 {
    range 192.0.2.20 192.0.2.100;
    option domain-name-servers 192.0.2.1;
    option routers 192.0.2.1;
    option broadcast-address 192.0.2.255;
    max-lease-time 172800;
}
```

这个示例为 `192.0.2.0/24` 网络添加了 **subnet** 声明。使用这个配置，DHCP 服务器会为发送这个子网的 DHCP 请求的客户端分配下列设置：

- **range** 参数定义的范围内空闲的 IPv4 地址
- 此子网的 DNS 服务器的 IP 为：**192.0.2.1**
- 此子网的默认网关为：**192.0.2.1**
- 此子网的广播地址为：**192.0.2.255**
- 最长租期时间，之后此子网中的客户端释放 IP，并向服务器发送新请求：**172800** 秒（2 天）

- 另外，配置在系统引导时自动启动 **dhcpd**：

```
# systemctl enable dhcpd
```

- 启动 **dhcpd** 服务：

■

```
# systemctl start dhcpd
```

- 对于 IPv6 网络：

1. 编辑 `/etc/dhcp/dhcpd6.conf` 文件：

- a. 另外，如果没有其他指令包含这些设置，则添加 `dhcpd` 用作默认值的全局参数：

```
option dhcp6.domain-search "example.com";
default-lease-time 86400;
```

本例将连接 `example.com` 的默认域名设为 **example.com**，默认的租期时间设为 **86400** 秒（1 天）。

- b. 在新行中添加 **authoritative** 语句：

```
authoritative;
```



重要

没有 **authoritative** 语句，如果客户端询问池外部的地址，则 **dhcpd** 服务不会回答带有 **DHCPNAK** 的 **DHCPREQUEST** 消息。

- c. 对于每个直接连接到服务器接口的 IPv6 子网，请添加 **subnet** 声明：

```
subnet6 2001:db8:0:1::/64 {
    range6 2001:db8:0:1::20 2001:db8:0:1::100;
    option dhcp6.name-servers 2001:db8:0:1::1;
    max-lease-time 172800;
}
```

本例为 `2001:db8:0:1::/64` 网络添加了一个 `subnet` 声明。使用这个配置，DHCP 服务器会为发送这个子网的 DHCP 请求的客户端分配下列设置：

- **range6** 参数中定义的范围内的 IPv6 地址。
- 此子网的 DNS 服务器的 IP 地址为 **2001:db8:0:1::1**。
- 最大租期时间是 **172800** 秒（2 天），之后此子网中的客户端释放 IP，并向服务器发送新请求。
请注意：IPv6 需要使用路由器广告信息来识别默认网关。

2. 另外，配置在系统引导时自动启动 **dhcpd6**：

```
# systemctl enable dhcpd6
```

3. 启动 **dhcpd6** 服务：

```
# systemctl start dhcpd6
```

其他资源

- [dhcp-options\(5\) 手册页](#)

- `dhcpd.conf` (5) 手册页
- `/usr/share/doc/dhcp-server/dhcpd.conf.example` file
- `/usr/share/doc/dhcp-server/dhcpd6.conf.example` file

3.9. 为没有直接连接到 DHCP 服务器的子网设置 DHCP 服务

如果 DHCP 服务器没有直接连接到该服务器应响应 DHCP 请求的子网，请使用以下步骤。如果 DHCP 中继代理将请求转发给 DHCP 服务器，则属于这种情况，因为 DHCP 服务器的接口没有一个直接连接到服务器应服务的子网。

根据您要为 IPv4、IPv6 或两个协议提供 DHCP 的信息，请查看以下操作过程：

- [IPv4 网络](#)
- [IPv6 网络](#)

先决条件

- 已以 `root` 用户身份登录。
- `dhcp-server` 软件包已安装。

步骤

- 对于 IPv4 网络：

1. 编辑 `/etc/dhcp/dhcpd.conf` 文件：

- a. 另外，如果没有其他指令包含这些设置，则添加 `dhcpd` 用作默认值的全局参数：

```
option domain-name "example.com";
default-lease-time 86400;
```

本例将连接 `example.com` 的默认域名设为 `example.com`，默认的租期时间设为 `86400` 秒（1 天）。

- b. 在新行中添加 `authoritative` 语句：

```
authoritative;
```



重要

没有 `authoritative` 语句，如果客户端询问池外部的地址，则 `dhcpd` 服务不会回答带有 `DHCPNAK` 的 `DHCPREQUEST` 消息。

- c. 为没有直接连接到服务器接口的 IPv4 子网添加 `shared-network` 声明，如下所示：

```
shared-network example {
    option domain-name-servers 192.0.2.1;
    ...

    subnet 192.0.2.0 netmask 255.255.255.0 {
```

```

range 192.0.2.20 192.0.2.100;
option routers 192.0.2.1;
}

subnet 198.51.100.0 netmask 255.255.255.0 {
range 198.51.100.20 198.51.100.100;
option routers 198.51.100.1;
}
...
}

```

这个示例添加了一个共享网络声明，其中包含对 192.0.2.0/24 和 198.51.100.0/24 的 **subnet** 声明。使用这个配置，DHCP 服务器会为发送来自这些子网之一 DHCP 请求的客户端分配下列设置：

- 两个子网的客户端的 DNS 服务器的 IP 是：**192.0.2.1**。
 - **range** 参数中定义范围内空闲的 IPv4 地址，具体取决于客户端从哪个子网发送请求。
 - 默认网关是 **192.0.2.1** 或 **198.51.100.1**，具体取决于客户端从哪个子网发送请求。
- d. 为服务器直接连接的子网添加 **subnet** 声明，用于访问上面 **shared-network** 中指定的远程子网：

```

subnet 203.0.113.0 netmask 255.255.255.0 {
}

```



注意

如果服务器没有向这个子网提供 DHCP 服务，则 **subnet** 声明必须为空，如示例中所示。没有对直接连接的子网的声明，**dhcpd** 不会启动。

2. 另外，配置在系统引导时自动启动 **dhcpd**：

```
# systemctl enable dhcpd
```

3. 启动 **dhcpd** 服务：

```
# systemctl start dhcpd
```

- 对于 IPv6 网络：

1. 编辑 **/etc/dhcp/dhcpd6.conf** 文件：

- a. 另外，如果没有其他指令包含这些设置，则添加 **dhcpd** 用作默认值的全局参数：

```

option dhcp6.domain-search "example.com";
default-lease-time 86400;

```

本例将连接 example.com 的默认域名设为 **example.com**，默认的租期时间设为 **86400** 秒（1 天）。

- b. 在新行中添加 **authoritative** 语句：

-


```
authoritative;
```



重要

没有 **authoritative** 语句，如果客户端询问池外部的地址，则 **dhcpcd** 服务不会回答带有 **DHCPNAK** 的 **DHCPREQUEST** 消息。

- c. 为没有直接连接到服务器接口的 IPv6 子网添加 **shared-network** 声明，如下所示：

```
shared-network example {
    option domain-name-servers 2001:db8:0:1::1:1
    ...

    subnet6 2001:db8:0:1::1:0/120 {
        range6 2001:db8:0:1::1:20 2001:db8:0:1::1:100
    }

    subnet6 2001:db8:0:1::2:0/120 {
        range6 2001:db8:0:1::2:20 2001:db8:0:1::2:100
    }
    ...
}
```

本例添加了一个共享网络声明，其中包含对 2001:db8:0:1::1:0/120 和 2001:db8:0:1::2:0/120 的 **subnet6** 声明。使用这个配置，DHCP 服务器会为发送来自这些子网之一 DHCP 请求的客户端分配下列设置：

- 两个子网的客户端的 DNS 服务器的 IP 是 **2001:db8:0:1::1:1**。
 - **range6** 参数中定义的空闲的 IPv6 地址，具体取决于客户端从哪个子网发送请求。请注意：IPv6 需要使用路由器广告信息来识别默认网关。
- d. 为服务器直接连接的子网添加 **subnet6** 声明，用于访问上面 **shared-network** 中指定的远程子网：

```
subnet6 2001:db8:0:1::50:0/120 {
}
```



注意

如果服务器没有向这个子网提供 DHCP 服务，则 **subnet6** 声明必须为空，如示例中所示。没有对直接连接的子网的声明，**dhcpcd** 不会启动。

2. 另外，配置在系统引导时自动启动 **dhcpcd6**：

```
# systemctl enable dhcpcd6
```

3. 启动 **dhcpcd6** 服务：

```
# systemctl start dhcpcd6
```

- [dhcp-options\(5\) 手册页](#)
- [dhcpd.conf \(5\) 手册页](#)
- [/usr/share/doc/dhcp-server/dhcpd.conf.example file](#)
- [/usr/share/doc/dhcp-server/dhcpd6.conf.example file](#)
- [设置 DHCP 转发代理](#)

3.10. 使用 DHCP 为主机分配静态地址

使用 **host** 声明，您可以配置 DHCP 服务器，来为主机的媒体访问控制（MAC）地址分配固定 IP 地址。例如：使用这个方法总是为服务器或者网络设备分配相同的 IP 地址。

根据您要为 IPv4、IPv6 或这两个协议配置固定地址，请查看以下操作过程：

- [IPv4 网络](#)
- [IPv6 网络](#)

前提条件

- **dhcpd** 服务已配置且正在运行。
- 已以 **root** 用户身份登录。

步骤

- 对于 IPv4 网络：
 1. 编辑 **/etc/dhcp/dhcpd.conf** 文件：
 - a. 添加一个 **host** 声明：

```
host server.example.com {  
    hardware ethernet 52:54:00:72:2f:6e;  
    fixed-address 192.0.2.130;  
}
```

这个示例将 DHCP 服务器配置为始终将 **192.0.2.130** IP 地址分配给 MAC 地址为 **52:54:00:72:2f:6e** 的主机。

dhcpd 服务根据 **fixed-address** 参数中指定的 MAC 地址识别系统，而不是根据 **host** 声明中的名称。因此，您可以将此名称设置为不与其它 **host** 声明匹配的任何字符串。要为多个网络配置相同的系统，请使用不同的名称，否则 **dhcpd** 无法启动。

- b. 另外，针对此主机的特定的 **host** 声明添加其他设置。

2. 重启 **dhcpd** 服务：

```
# systemctl start dhcpd
```

- 对于 IPv6 网络：
 1. 编辑 **/etc/dhcp/dhcpd6.conf** 文件：

- a. 添加一个 **host** 声明：

```
host server.example.com {
    hardware ethernet 52:54:00:72:2f:6e;
    fixed-address6 2001:db8:0:1::200;
}
```

这个示例将 DHCP 服务器配置为始终将 **2001:db8:0:1::20** IP 地址分配给 MAC 地址为 **52:54:00:72:2f:6e** 的主机。

dhcpcd 服务根据 **fixed-address6** 参数中指定的 MAC 地址识别系统，而不是根据 **host** 声明中的名称。因此，您可以将此名称设置为任何字符串，只要它对于其它 **host** 声明是唯一的。要为多个网络配置相同的系统，请使用不同的名称，因为，否则的话 **dhcpcd** 无法启动。

- b. 另外，对针对此主机的特定的 **host** 声明添加其他设置。

2. 重启 **dhcpcd6** 服务：

```
# systemctl start dhcpcd6
```

其他资源

- [dhcpc-options\(5\)](#) 手册页
- `/usr/share/doc/dhcp-server/dhpcpd.conf.example` file
- `/usr/share/doc/dhcp-server/dhpcpd6.conf.example` file

3.11. 使用 **GROUP** 声明同时将参数应用到多个主机、子网和共享网络

使用 **group** 声明，您可以将同样的参数应用到多个主机、子网和共享网络。

请注意，流程描述了对主机使用 **group** 声明，但步骤与子网和共享网络的相同。

根据您要为 IPv4、IPv6 或两个协议配置组，请查看以下操作过程：

- [IPv4 网络](#)
- [IPv6 网络](#)

先决条件

- **dhcpcd** 服务已配置且正在运行。
- 已以 **root** 用户身份登录。

步骤

- 对于 IPv4 网络：
 1. 编辑 `/etc/dhcp/dhpcpd.conf` 文件：
 - a. 添加一个 **group** 声明：

```
group {
    option domain-name-servers 192.0.2.1;

    host server1.example.com {
        hardware ethernet 52:54:00:72:2f:6e;
        fixed-address 192.0.2.130;
    }

    host server2.example.com {
        hardware ethernet 52:54:00:1b:f3:cf;
        fixed-address 192.0.2.140;
    }
}
```

这个 **group** 定义对两个 **host** 条目进行分组。**dhcpd** 服务将 **option domain-name-servers** 参数中设置的值应用到组中的两个主机。

b. 另外，对特定于这些主机的 **group** 声明添加其他设置。

2. 重启 **dhcpd** 服务：

```
# systemctl start dhcpd
```

- 对于 IPv6 网络：

1. 编辑 **/etc/dhcp/dhcpd6.conf** 文件：

a. 添加一个 **group** 声明：

```
group {
    option dhcp6.domain-search "example.com";

    host server1.example.com {
        hardware ethernet 52:54:00:72:2f:6e;
        fixed-address 2001:db8:0:1::200;
    }

    host server2.example.com {
        hardware ethernet 52:54:00:1b:f3:cf;
        fixed-address 2001:db8:0:1::ba3;
    }
}
```

这个 **group** 定义对两个 **host** 条目进行分组。**dhcpd** 服务将 **option dhcp6.domain-search** 参数中设置的值应用到组中的两个主机。

b. 另外，对特定于这些主机的 **group** 声明添加其他设置。

2. 重启 **dhcpd6** 服务：

```
# systemctl start dhcpd6
```

其他资源

- [dhcp-options\(5\)](#) 手册页

- `/usr/share/doc/dhcp-server/dhcpd.conf.example` file
- `/usr/share/doc/dhcp-server/dhcpd6.conf.example` file

3.12. 恢复损坏的租期数据库

如果 DHCP 服务器记录了一个与租期数据库有关的错误，如 **Corrupt lease file - possible data loss!**，则您可以从 `dhcpd` 服务创建的副本中恢复租期数据库。请注意，这个副本可能没有反映数据库的最新状态。



警告

如果您删除了租期数据库而不是用备份替换它，则丢失了当前分配的租期的所有信息。因此，DHCP 服务器可以为之前分配给其它主机但还没有过期的客户端分配租期。这会导致 IP 冲突。

根据您要恢复 DHCPv4、DHCPv6 还是两个数据库，请查看：

- [恢复 DHCPv4 租期数据库](#)
- [恢复 DHCPv6 租期数据库](#)

先决条件

- 已以 `root` 用户身份登录。
- 租期数据库被损坏。

流程

- 恢复 DHCPv4 租期数据库：

1. 停止 `dhcpd` 服务：

```
# systemctl stop dhcpd
```

2. 重命名损坏租期数据库：

```
# mv /var/lib/dhcpd/dhcpd.leases /var/lib/dhcpd/dhcpd.leases.corrupt
```

3. 恢复 `dhcp` 服务在刷新租期数据库时创建的租期数据库的副本：

```
# cp -p /var/lib/dhcpd/dhcpd.leases~ /var/lib/dhcpd/dhcpd.leases
```



重要

如果您有租期数据库的最新备份，则恢复这个备份。

4. 启动 **dhcpd** 服务：

```
# systemctl start dhcpd
```

- 恢复 DHCPv6 租期数据库：

1. 停止 **dhcpd6** 服务：

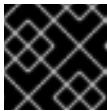
```
# systemctl stop dhcpd6
```

2. 重命名损坏租期数据库：

```
# mv /var/lib/dhcpd/dhcpd6.leases /var/lib/dhcpd/dhcpd6.leases.corrupt
```

3. 恢复 **dhcp** 服务在刷新租期数据库时创建的租期数据库的副本：

```
# cp -p /var/lib/dhcpd/dhcpd6.leases~ /var/lib/dhcpd/dhcpd6.leases
```



重要

如果您有租期数据库的最新备份，则恢复这个备份。

4. 启动 **dhcpd6** 服务：

```
# systemctl start dhcpd6
```

其他资源

- [dhcpd 服务的租期数据库](#)

3.13. 设置 DHCP 转发代理

DHCP 中继代理(**dhcrelay**)可以将来自没有 DHCP 服务器的子网的 DHCP 和 BOOTP 请求中继到其他子网上的一个或多个 DHCP 服务器。当 DHCP 客户端请求信息时，DHCP 转发代理会将该请求转发到指定的 DHCP 服务器列表。当 DHCP 服务器返回一个回复时，DHCP 转发代理会将此请求转发给客户端。

根据您要为 IPv4、IPv6 或两个协议设置 DHCP 转发，请查看以下操作过程：

- [IPv4 网络](#)
- [IPv6 网络](#)

先决条件

- 已以 **root** 用户身份登录。

步骤

- 对于 IPv4 网络：
 1. 安装 **dhcp-relay** 软件包：

```
# yum install dhcp-relay
```

2. 将 `/lib/systemd/system/dhcrelay.service` 文件复制到 `/etc/systemd/system/` 目录中：

```
# cp /lib/systemd/system/dhcrelay.service /etc/systemd/system/
```

不要编辑 `/usr/lib/systemd/system/dhcrelay.service` 文件。`dhcp-relay` 软件包的将来更新可能会覆盖更改。

3. 编辑 `/etc/systemd/system/dhcrelay.service` 文件，并追加 `-i interface` 参数以及负责该子网的 DHCPv4 服务器的 IP 地址列表：

```
ExecStart=/usr/sbin/dhcrelay -d --no-pid -i enp1s0 192.0.2.1
```

使用这些额外的参数，`dhcrelay` 会侦听 `enp1s0` 接口上的 DHCPv4 请求，并将它们转发到 IP 为 `192.0.2.1` 的 DHCP 服务器。

4. 重新加载 `systemd` 管理器配置：

```
# systemctl daemon-reload
```

5. （可选）配置在系统引导时启动 `dhcrelay` 服务：

```
# systemctl enable dhcrelay.service
```

6. 启动 `dhcrelay` 服务：

```
# systemctl start dhcrelay.service
```

- 对于 IPv6 网络：

1. 安装 `dhcp-relay` 软件包：

```
# yum install dhcp-relay
```

2. 将 `/lib/systemd/system/dhcrelay.service` 文件复制到 `/etc/systemd/system/` 目录中，并将其命名为 `dhcrelay6.service`：

```
# cp /lib/systemd/system/dhcrelay.service /etc/systemd/system/dhcrelay6.service
```

不要编辑 `/usr/lib/systemd/system/dhcrelay.service` 文件。`dhcp-relay` 软件包的将来更新可能会覆盖更改。

3. 编辑 `/etc/systemd/system/dhcrelay6.service` 文件，并追加 `-l receiving_interface` 和 `-u outgoing_interface` 参数：

```
ExecStart=/usr/sbin/dhcrelay -d --no-pid -l enp1s0 -u enp7s0
```

使用这些额外的参数，`dhcrelay` 会侦听 `enp1s0` 接口上的 DHCPv6 请求，并将它们转发给连接到 `enp7s0` 接口的网络。

4. 重新加载 `systemd` 管理器配置：

```
# systemctl daemon-reload
```

5. (可选) 配置在系统引导时启动 **dhcrelay6** 服务 :

```
# systemctl enable dhcrelay6.service
```

6. 启动 **dhcrelay6** 服务 :

```
# systemctl start dhcrelay6.service
```

其他资源

- **dhcrelay(8)** 手册页