DNS 基础架构实验

1 概述

DNS（域名系统，Domain Name System）是互联网的电话簿；它将主机名转换为 IP 地址（反之亦然）。这种转换过程通过 DNS 系统在幕后自动完成。解析过程涉及许多名称服务器，包括根服务器、顶级域（Top-Level Domain, TLD）服务器和最终域服务器。这些名称服务器共同构成了整个 DNS 基础架构，它们确保了互联网上所有设备之间的通信顺畅无阻。

为了帮助学生理解这些名称服务器如何协同工作以形成该基础设施，我们将在一个互联网模拟器中创建一个小规模的 DNS 系统。虽然这个小规模的系统很小，但它拥有真实 DNS 基础设施的所有必要元素。通过建立这样一个系统，学生们将对 DNS 的实际运作有更深的理解。虽然本实验不是安全实验的一部分，但它是几个 SEED 实验的基础。本实验涵盖以下主题：

- DNS 基础架构
- DNS 正向和反向查找过程
- 根服务器和顶级域（Top-Level Domain, TLD）服务器
- DNS 解析器、本地 DNS 服务器

阅读材料与视频。 关于 DNS 协议的详细内容可以在以下资源中找到：

- SEED 书籍第 23 章，*Computer & Internet Security: A Hands-on Approach*, 3rd Edition, by Wenliang Du. 详情请见 `https://www.handsonsecurity.net`.
- SEED 书籍第 10 章，*Internet Security: A Hands-on Approach*, 3rd Edition, by Wenliang Du. 详情请见 `https://www.handsonsecurity.net`.
- SEED 讲义第七部分，*Internet Security: A Hands-on Approach*, by Wenliang Du. 详情请见 `https://www.handsonsecurity.net/video.html`.

实验环境。 本实验在我们预先构建好的 Ubuntu 20.04 VM（可以从我们的 SEED 网站当中下载）当中测试可行。既然我们使用容器来建立实验环境，本实验不太依赖 SEED VM。您可以在其他 VM、物理机器以及云端 VM 上进行此实验。

实验任务概述。 DNS 基础设施涉及许多服务器。我们将在一个模拟器中构建简化后的 DNS 基础架构。这个基础设施包括两个根服务器、两个 `com` TLD (Top-Level Domain) 服务器，一个 `edu` TLD 服务器以及两个二级域名名称服务器 (见图 1)。注意学生需要将 `smith2021` 替换为自己的姓氏和当前年份

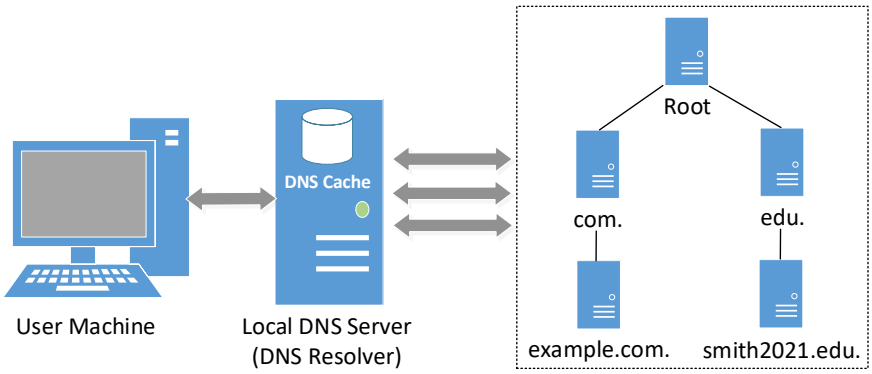

图 1: 简化后的 DNS 基础架构

我们需要为图中所示的每个区域配置名称服务器。我们将从底部向上进行配置。具体来说，首先配置二级域名名称服务器（任务一：配置二级域名服务器），然后是 TLD 服务器（任务二：配置顶级域服务器），接着是根服务器（任务三：配置根服务器）。我们还需要配置本地 DNS 服务器和客户端机器（任务四和五），这样客户端就可以使用这个 DNS 系统了。任务六是完善支持反向解析功能的 DNS 系统。

2 实验设置与 SEED Internet 仿真器

本实验将在 SEED Internet 仿真器内进行（本文档中简称“仿真器”）。如果这是您第一次使用仿真器，那么阅读本节非常重要。我们建议教师提供一个实验室课程来帮助学生熟悉仿真器。

2.1 Internet 仿真器

我们提供两种形式的预构建仿真器：Python 代码和容器文件。容器文件由 Python 代码生成；运行 Python 代码需要从 GitHub 安装 SEED 仿真器源代码，而容器文件无需安装仿真器源代码即可直接使用。希望定制仿真器的教师可以修改 Python 代码、生成自己的容器文件，并用它们替换实验设置文件中包含的文件。具体说明请参阅 `README.md`。

下载仿真器文件。 请从实验网页下载 `Labsetup.zip` 并解压。容器文件位于实验设置目录的 `output/`文件夹中；如果一个实验包含多个仿真器，文件夹名称可能有所不同，具体名称会在实验任务中说明。

启动仿真。 进入容器文件夹，运行以下 Docker Compose 命令构建并启动容器。建议在 SEED Ubuntu 20.04 虚拟机中运行；在安装了 Docker 的普通 Ubuntu 系统中也可以运行。

```
$ docker-compose build        # 构建容器镜像
$ docker-compose up           # 启动容器
$ docker-compose down         # 关闭容器

// SEED虚拟机中提供的别名
$ dcbuild
```

```
$ dcup
$ dcdown
```

所有容器都在后台运行。要进入容器执行命令，可以先使用 `docker ps` 查找容器 ID，再使用 `docker exec` 启动 shell。SEED 虚拟机为这些命令提供了以下别名：

```
$ dockps                    # docker ps的别名
$ docksh <id>               # 在指定容器中启动shell

$ dockps
b1004832e275  hostA-10.9.0.5
0af4ea7a3e2e  hostB-10.9.0.6
9652715c8e0a  hostC-10.9.0.7
$ docksh 96
root@9652715c8e0a:/#
```

容器 ID 只需输入足以唯一识别容器的前几个字符。如果实验环境启动失败，请参阅容器手册中的“常见问题”部分。

设置终端标题。 实验中通常需要同时打开多个容器终端。进入容器后，可以用以下任一命令设置终端标签页的标题：

```
# set_title New-Title
# st New-Title
```

2.2 仿真的 Internet 地图

仿真器中的每台计算机（主机或路由器）都是一个 Docker 容器。用户可以通过 Docker 命令访问这些计算机，例如在容器中启动 shell。仿真器还提供了一个 Web 应用程序，用于显示所有主机、路由器和网络。

启动仿真器后，可以通过`http://localhost:8080/map.html`访问网络地图，如图 2所示。使用鼠标滚轮可以缩放地图；单击任意节点会在侧边栏中显示该节点的详细信息，并可从侧边栏打开该节点的容器终端。

2.3 过滤和回放

用户可以设置过滤器来可视化网络流量。过滤器语法与 `tcpdump` 相同；过滤表达式会直接传递给所有节点上运行的 `tcpdump` 程序。当节点观察到符合条件的数据包时，它会向网络地图发送事件，地图会短暂高亮该节点。

有时一系列事件发生得太快，难以观察其先后顺序。此时可以使用回放面板记录事件，再以较慢速度重放。通过调整事件间隔可以改变回放速度。图 3展示了数据包过滤器和事件回放控件。

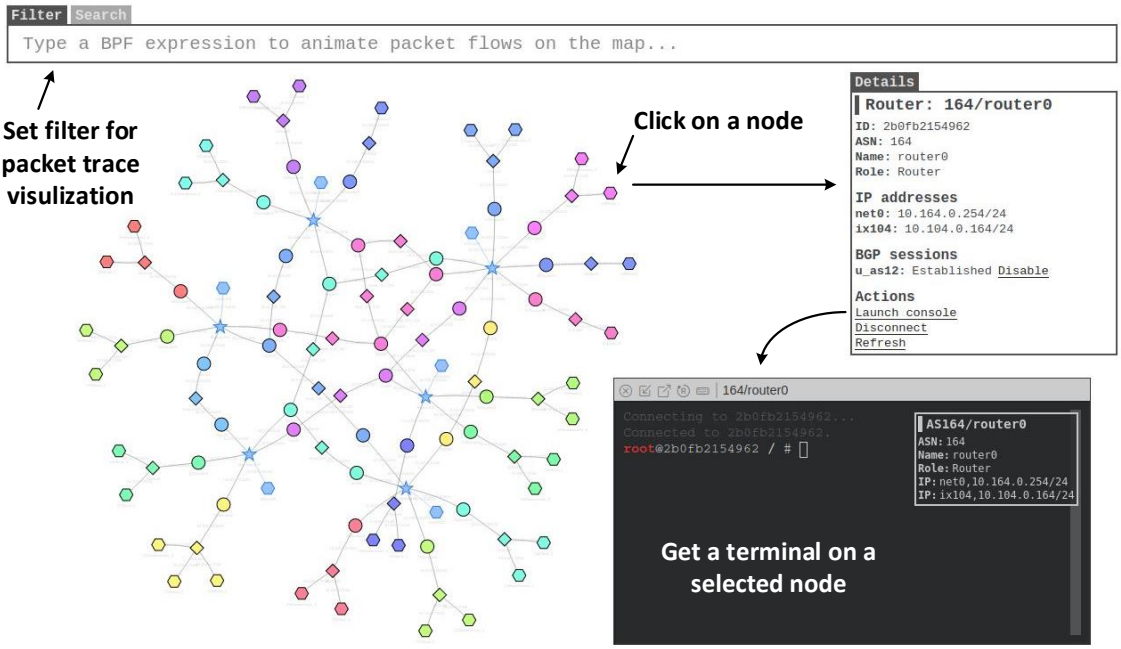

图 2: 仿真的 Internet 地图

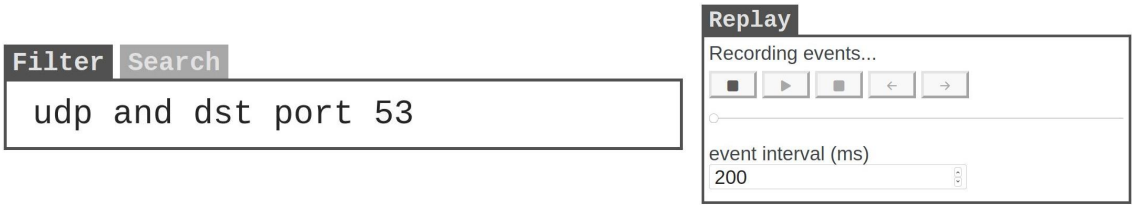

图 3: 捕获并回放网络事件

2.4 所有的名称服务器

在这个仿真器中，DNS 基础设施由多个运行 DNS 名称服务器的容器组成。我们已经自定义了它们的容器名，以便于查找。所有容器的名字都包含术语 `DNS`，并且他们的 IP 地址也包含在名字里。以下命令可以列出所有的这些容器（为了显示目的重新排列了顺序）。

```
$ dockps | grep DNS
dedc6676421e  as150h-DNS-Root-A-10.150.0.72
de5dc343224f  as160h-DNS-Root-B-10.160.0.72
84cf7c7f337d  as151h-DNS-COM-A-10.151.0.72
20c134dceee4  as161h-DNS-COM-B-10.161.0.72
f20fe7ed115f  as152h-DNS-EDU-10.152.0.71
cfabce676bed  as154h-DNS-Example-10.154.0.71
5801404d45d2  as162h-DNS-AAAAA-10.162.0.72
```

```
c357ff76bda6  as153h-Global_DNS-1-10.153.0.53
d65e76c44437  as163h-Global_DNS-2-10.163.0.53

// Get a shell on a selected container
$ docksh cfab   ← container ID
root@cfabce676bed / #
root@cfabce676bed / # st Example    ← change the terminal title
```

2.5 复制 DNS 区域文件

在这个实验中，我们需要修改几个容器内的 DNS 区域文件（以及一些配置文件）。我们可以在这些容器内进行修改。问题是，一旦容器停止并删除后，那些更改就会丢失。最好将这些文件复制到主机上，在主机上进行修改后再将其复制回容器。我们可以使用"docker cp" 命令来完成此操作。由于我们需要频繁运行此命令，所以编写了以下 Shell 脚本，一个用于从容器获取区域文件到主机，另一个用于将区域文件复制回容器。这两个文件都包含在实验设置文件夹中。

Listing 1: 从容器获取区域文件: getzone.sh

```
#!/bin/bash

keyword=$1   # Command-line argument 1: used to identify container
filename=$2  # Command-line argument 2: zone file name on container
filename_to=${keyword}_${filename}                                    ①
containerID=$(docker ps | grep $keyword | awk '{print $1}')          ②
if [[ ! -f $filename_to ]]; then
   docker cp $containerID:/etc/bind/zones/$filename ./$filename_to   ③
else
   echo "** File $filename_to already exists; will not overwrite."
fi
```

在仿真器中，区域文件存储在 /etc/bind/zones 文件夹内。每个 DNS 名称服务器的容器名中有唯一的关键词，例如，处理 example.com 区域的名称服务器其容器名中含有关键词 Example。通过使用此关键词，我们可以找到对应容器的唯一标识符（见行～② ）。同时在区域文件名前添加该关键词，以便于一旦复制到主机上时能够轻松识别它们（见行 ① 和 ③ ）。以下示例演示了如何从两个根名称服务器和 example.com 名称服务器获取区域文件。

```
$ ./getzone.sh Root-A root           ←  New file: Root-A_root
$ ./getzone.sh Root-B root           ←  New file: Root-B_root
$ ./getzone.sh Example example.com.  ←  New file: Example_example.com
```

将文件复制回容器的过程类似，只是在区域文件被复制后需要重启名称服务器（见行 ④ ）。

Listing 2: 向容器发送区域文件: sendzone.sh

```
#!/bin/bash
```

```
keyword=$1   # Command-line argument 1: used to identify container
filename=$2  # Command-line argument 2: zone file name on container
filename_from=${keyword}_${filename}
containerID=$(docker ps | grep $keyword | awk '{print $1}')
if [[ -f $filename_from ]]; then
   echo "== Copy zone file to container"
   docker cp ./$filename_from $containerID:/etc/bind/zones/$filename

   echo "== Restart the nameserver"
   docker exec $containerID service named restart      ④
else
   echo "** File $filename_from does not exists."
fi
```

下面是使用 sendzone.sh 的示例。它将 Root-A 的区域文件（Root-A_root）发送回 Root-A 容器，并重启名称服务器。

```
$ ./sendzone.sh Root-A root
== Copy zone file to container
== Restart the nameserver
  * Stopping domain name service... named
waiting for pid 92 to die
    ...done.
  * Starting domain name service... named
    ...done.
```

3 任务 1：配置域名服务器

在这个任务中，我们将为两个域配置名称服务器，一个是 example.com ，另一个是基于学生姓名自定义的名称。

3.1 任务 1.a：配置 example.com 的名称服务器

在这个任务中，我们需要在仿真器内一个容器中的域名 example.com 上配置名称服务器。该容器名包含关键词 Example ，以及 IP 地址。

```
$ dockps | grep Example
46ab89e26738  as154h-DNS-Example-10.154.0.71
```

步骤 1：添加区域条目。 要为 example.com 域托管服务器，我们需要在 BIND 的配置中添加一个区域条目，以便它知道将要托管哪些区域。在仿真器中，名称服务器托管的区域位于路径 /etc/bind/named.conf.zones 文件内（可以有多个区域）。以下是一个示例，在此示例中，file 条目指定了包含实际区域信息的实际文件。如果我们想在此名称服务器上托管另一个区域，可以在该文件中添加相应

的区域条目。

```
zone "example.com" {
    type master;         ← this is the master server
    allow-update { any; };
    file "/etc/bind/zones/example.com.";  ← the actual zone file
};

zone "example.net" {
    ...
}
```

上面 example.com 的区域条目表示当前名称服务器是该域名的主服务器，并且在 file 条目中指定了实际的区域文件。在仿真器中，区域文件位于路径 /etc/bind/zones 文件夹内。需要注意的是，除了根区域外，大多数区域文件（filename）以点（.）结尾。这是本实验中需要修改的主要文件。

步骤 2：修改区域文件。 向区域文件添加至少四个记录（类型为 A ），以便将主机名映射到 IP 地址。您可以使用星号 (*) 作为通配符主机名。

```
$TTL 300               ← The default time to live: 300 seconds
$ORIGIN example.com.   ← The start of this zone file in the namespace
@ SOA ns1.example.com. admin.example.com. 1635647622 900 900 1800 60

@                  NS    ns1.example.com.
ns1.example.com.   A     10.154.0.71

www                A     10.154.0.72
abc                A     10.154.0.73
```

区域文件必须指定权威主名称服务器的起始授权（SOA）记录，包括该域名的权威主名称服务器名和负责管理该名称服务器的电子邮件地址。SOA 记录还指定了一个过期时间参数列表（序列号、从属刷新周期、从属重试时间、从属过期时间和缓存记录的最大时间）。如果我们修改区域文件，应改变序列号 (SOA 条目中的高亮数字)，以使更改可以同步到从服务器。

在区域文件中, 以全点字符（即句点）结尾的域名是完全限定的；而没有以句点结尾的则是相对于当前起点而言的。例如，在上面的例子中，ns1.example.com. 是一个完整名称，而 www 表示的是 www.example.com。

步骤 3：重启名称服务器。 每次修改 BIND 配置文件后, 我们都需要重新启动名称服务器以使更改生效。如果配置文件有误，名称服务器将无法正常启动。

```
# service named restart
 * Stopping domain name service... named
          waiting for pid 92 to die

    [ OK ]
```

```
 * Starting domain name service... named
```

步骤 4：测试。 由于我们还没有完成整个 DNS 基础设施的设置, 我们需要使用 `@10.154.0.71` 选项将我们的 DNS 查询直接发送到 `10.154.0.71`, 这是我们在仿真器内 `example.com` 名称服务器的 IP 地址。如果操作无误, 您可以得到您在区域文件中指定的 IP 地址。请报告您的观察结果。

```
$ dig @10.154.0.71 www.example.com
...
;; ANSWER SECTION:
xyz.example.com.    300    IN    A    1.2.3.6
...
```

3.2 任务 1.b：配置另一个域的名称服务器

在这个任务中，我们将为另一个域配置名称服务器。该域名应采用以下格式 `<NAME><YEAR>.edu`，其中 `<NAME>` 应替换为您的姓氏，而 `<YEAR>` 用当前年份替代。例如, 如果您的姓是 `Smith`, 当前的年份是 2021 年，则域名将是 `smith2021.edu`。在仿真器中，已创建了一个空名称服务器；容器名包含关键词 `AAAAA` 请使用此名称服务器来托管您的域。

```
$ dockps | grep AAAAA
70f2e8a57a96  as162h-DNS-AAAAA-10.162.0.72
```

4 任务 2：配置 TLD 服务器

在这个任务中，我们将为两个 TLD 域名 `com` 和 `edu` 配置名称服务器。我们已经创建了三个名称服务器来托管这两个 TLD 域名。

```
$ dockps | grep DNS
c3040914667c  as151h-DNS-COM-A-10.151.0.72    ← The master com server
3fa8fa41afb6  as161h-DNS-COM-B-10.161.0.72    ← The slave com server
6c7132feb7d4  as152h-DNS-EDU-10.152.0.71      ← The edu server
...
```

TLD `com` 区域有两个名字服务器。一个配置为主服务器，另一个为从属服务器。我们只需要修改主服务（COM-A）的区域文件, 因为从属服务器会自动与主服务器同步。以下是在这两个服务器上的区域配置。

```
// For master (COM-A)
zone "com." { type master; allow-transfer { any; }; ...};

// For slave (COM-B)
zone "com." { type slave; masters { 10.151.0.72; }; ... };
```

TLD 域内的所有名称服务器都必须在该 TLD 服务器中注册它们的名称服务, 否则任何人都找不到它们。对于每个域名，如 example.com，需要向 TLD com 的区域文件添加两个条目：一个 NS 记录和一个 A 记录。NS 记录指定 example.com 域名的名称服务器，A 记录指定名称服务器的 IP 地址。

实验任务。 请将您的 example.com 和 <NAME><YEAR>.edu 名称服务器注册到其相应的 TLD 服务器。记得在更改后运行"service named restart" 以重启名称服务器。同时需要改变区域文件的序列号；否则，主服务器上的更改不会同步到从属服务器上。完成更改后，请执行以下命令测试服务并报告您的观察结果。

```
# dig @10.151.0.72 www.example.com       ← Query the COM-A server
# dig @10.161.0.72 www.example.com       ← Query the COM-B server
# dig @10.152.0.72 www.<NAME><YEAR>.edu  ← Query the EDU server
```

需要注意的是，TLD 服务器被配置为不执行递归查询，因此只会告诉您域名查询的名称服务器；它不会为您解析该查询。见下方配置。

```
// File: named.conf.options
options {
        directory "/var/cache/bind";
        recursion no;
        dnssec-validation no;
        empty-zones-enable no;
        allow-query { any; };
        allow-update { any; };
};
```

5 任务 3：配置根服务器

在这个任务中，我们将为根区域配置名称服务器。在现实世界中，有 13 个给根区的名称服务器，它们通过 IANA 维护的根区域文件进行同步。在我们的仿真器里，我们创建了两个根服务器，并且其容器名包含关键词 Root 。我们将通过将相同的文件内容放入它们的区域文件中来手动使它们同步。

```
$ dockps | grep Root
9a96092c7c85  as150h-DNS-Root-A-10.150.0.72
60f171f92287  as160h-DNS-Root-B-10.160.0.72
```

所有 TLD 名称服务器都需要注册到根服务器，以便在 DNS 查询过程中能够找到它们。对于每一个我们想包含在微型 DNS 系统中的 TLD 区域，我们需要在其区域文件中至少添加两个记录，包括一个 NS 记录和一个 A 记录。在此任务中，学生需要修改这两个根服务器的区域文件以支持仿真器中的 com 和 edu TLDs。区域文件位于路径 /etc/bind/zones 文件夹内。在完成更改后，请重启两个名称服务器，并从另一个容器执行以下命令（用根服务器 IP 地址替换 <root-ip>；请尝试两个根服务器）。报告并解释您的观察结果。

```
$ dig @<root-ip>    <ANYNAME>.com   ← choose an arbitrary name
```

```
$ dig @<root-ip>    <ANYNAME>.edu
$ dig @<root-ip>    com
$ dig @<root-ip>    edu
```

6 任务 4：配置本地 DNS 服务器

当计算机需要从主机名解析 IP 地址（反之亦然）时，它会向其助手发送请求，该助手被称为本地 DNS 服务器或 DNS 解析器。这个 DNS 解析器将进行整个 DNS 解析过程，并然后将结果返回给计算机。尽管称之为“本地”，但此服务器并不一定位于本地位置。在过去，当设置计算机时，通常使用自己的本地网络上的 DNS 服务器，但现在，有许多非本地的 DNS 服务器可以作为“本地”DNS 服务器来使用。例如，Google 公共 DNS 就是由 Google 提供的一项 DNS 服务，可供 Internet 上任何主机使用。

当我们配置根、TLD 和域名服务器时，我们将其设置为非递归的，即它们只会告诉您他们所知道的信息，而不会进行整个解析过程以获取最终答案。当我们将本地 DNS 服务器配置为开启递归选项（如下所示），它会为您获取答案。

```
options {
    recursion yes;
    ...
};
```

如果本地 DNS 服务器在缓存中找不到答案，它将通过迭代过程从外部名称服务器获取答案。这个过程从根服务器开始。因此，本地 DNS 服务器需要知道根服务器的 IP 地址。

在路径 `/etc/bind/named.conf.default-zones` 文件内（包含于 BIRD 配置文件 `/etc/bind/named.conf` 中）有一个为根区域设置的条目。这个条目指定了一个根区域的提示文件，这就是使本地 DNS 服务器知道根服务器 IP 地址的源头。

```
zone "." {
    type hint;
    file "/usr/share/dns/root.hints";
};
```

在现实世界中，有 13 个给根区的名称服务器，因此它们的 IP 地址被提供在 `root.hints` 文件内。在我们的仿真器里，我们只有两个根服务器。他们的信息已经添加到提示文件内。

```
.       NS   ns1.
.       NS   ns2.
ns1.    A    10.150.0.72
ns2.    A    10.160.0.72
```

任务。 我们在仿真器内创建了两个 DNS 解析器。我们已经在仿真的容器名中加入了关键词 `Global_DNS` 并且包含了解析器的 IP 地址。您可以使用以下命令找到这两个 DNS 解析器。

```
$ dockps | grep Global_DNS
```

```
c357ff76bda6  as153h-Global_DNS-1-10.153.0.53
d65e76c44437  as163h-Global_DNS-2-10.163.0.53
```

请到他们的 root.hints 文件。在文件内，请将根名称服务器的 IP 地址更改其它地址（例如，把 72 改成 75），然后展示这如何影响 DNS 解析过程（执行以下命令）。请报告并解释您的观察结果。请记得完成此任务后将其改回，因为后续的任务依赖于它们。

```
# dig @10.153.0.53  example.com
# dig @10.163.0.53  example.com
```

7 任务 5：配置客户端

到目前为止，我们需要在我们的 dig 命令中使用 @<ip> 来指示 dig 命令应该与哪个 DNS 服务器通信。虽然这对 dig 命令不是问题, 但对于依赖于 DNS 的其他软件来说这是一个问题。我们需要告诉操作系统它应该使用哪个 DNS 服务器。我们可以通过在用户的机器上更新解析器的配置文件 /etc/resolv.conf 文件来指示操作系统使用特定的本地 DNS 解析器。在文件的第一个 nameserver 条目里填入容器的 IP 地址，即该服务器将作为主要的 DNS 解析器。

我们已经配置了所有机器使用 10.153.0.53 作为主要 DNS 解析器，除了属于 AS-155 的机器。你的任务是配置 AS-155 中的主机，使它们能够使用其中一个（或两个）DNS 解析器。你可以在 /etc/resolv.conf 文件中添加如下条目之一（或两个）：

```
nameserver 10.153.0.53
nameserver 10.163.0.53
```

测试整个 DNS 基础设施。 此时，DNS 基础设施已经完全设置好，我们可以运行 dig 而不使用 @<ip> 选项；dig 将使用系统设置的本地 DNS 服务器。请从 AS-155 中的一个主机上执行以下命令。如果一切配置正确，你应该能够得到预期的答案。

```
# dig www.example.com
# dig <NAME><YEAR>.edu
```

请使用地图显示数据包跟踪情况。首先设置过滤器为 "udp and dst port 53" 来捕获 DNS 请求流量。然后执行这些 dig 命令。如果整个过程发生得太快，你可以使用地图的重播功能：在设置好过滤器后，点击记录按钮，再执行上述 dig 命令。停止录制并播放所录制的事件。如有必要，请调整时间间隔。因为截取数据包追踪的屏幕截图有些困难，所以在报告中描述数据包追踪情况就足够了。

8 任务 6：反向 DNS 查找

反向 DNS 查找是从 IP 地址找出主机名的过程。这一过程实际上与正向查找非常相似。我们用一个例子来说明此过程。给定一个 IP 地址，例如 128.230.171.184 这个 IP 地址属于 syr.edu，DNS 解析器构造了一个“伪造”的名称 184.171.230.128.in-addr.arpa，然后通过一个迭代过程发送查询请

求，就像正向查找一样。也就是说，它从根服务器开始，到 in-addr.arpa 服务器、128.in-addr.arpa 服务器，最终到达托管 syr.edu 区域的 230.128.in-addr.arpa 服务器。

任务 1 到 4 只设置了正向查找所需的 DNS 基础设施。在这项任务中，我们将设置反向查找所需的基础设施。出于简化的考量，我们仅支持仿真器中的 example.com 域 IP 地址的反向查询（即，在 10.154.0.0/24 网络中的 IP 地址）。你需要设置以下名称服务器：

- 步骤 1：配置根服务器。根服务器需要托管 in-addr.arpa 域的 NS 记录。我们需要在两台根服务器的 /etc/bind/zones/root 文件里分别添加相应的名称服务器记录。
- 步骤 2：设置 in-addr.arpa 服务器。该服务器应托管 in-addr.arpa. 区域。我们可以使用 com 域名服务器来托管这个区域，通过在/etc/bind/named.conf.zones 文件中添加该区域实现。一个域名服务器可以同时托管多个不同的区域。

```
zone "com." {
        ... already there ...
};

zone "in-addr.arpa." {
        type master;
        notify yes;
        allow-transfer { any; };
        allow-update { any; };
        file "/etc/bind/zones/in-addr.arpa.";  ← the zone file
};
```

我们需要创建上述条目中指定的区域文件 in-addr.arpa.。我们在该区域文件中添加 154.10.in-addr.arpa 区域的 NS 记录。为了简化，我们直接为 154.10 子区域提供 NS 记录，而不是只为 10.in-addr.arpa 区域提供 NS 记录（然后，依赖 10.in-addr.arpa 服务器来提供 154.10 子区域的信息）。下面是一个简化的区域文件示例：

```
$TTL 300
$ORIGIN in-addr.arpa.

@        SOA ns1.com. admin.com. 1565237345 900 900 1800 60
@        NS  ns1.com.
@        NS  ns2.com.
ns1.com. A 10.151.0.72
ns2.com. A 10.161.0.72

; Please replace ____ with the correct information
154.10.in-addr.arpa.  NS  ____   ← The nameserver of the domain
____                  A   ____   ← The IP address of the nameserver
```

- 步骤 3：配置 154.10.in-addr.arpa 服务器：此域名服务器可以在托管 example.com 区域的同一域名服务器上运行。我们应该在其/etc/bind/named.conf.zones 文件中添加一个区域条目（类

似于步骤 2），然后创建相应的区域文件，该文件应包含 example.com 域的逆向解析记录。以下是一个可能的区域文件示例：

```
$TTL 300

$ORIGIN 154.10.in-addr.arpa.
@ SOA ns1.example.com. admin.example.com. 1635647622 900 900 1800 60

@               NS ns1.example.com.
ns1.example.com. A  10.154.0.71

71.0     IN      PTR     ns1.example.com.
72.0     IN      PTR     www.example.com.
73.0     IN      PTR     abc.example.com.
```

测试。 在对名称服务器进行修改后，请确保运行 "service named restart" 来重启 DNS 服务器。如果一切设置正确，我们可以到模拟器中的任意主机上运行以下反向查询命令。请记录并说明您的观察结果。请使用 map 展示其中一个命令的数据包路径，并在报告中描述该数据包路径。

```
dig -x 10.154.0.71
dig -x 10.154.0.72
dig -x 10.154.0.73
```

9 提交

你需要提交一份带有截图的详细实验报告来描述你所做的工作和你观察到的现象。你还需要对一些有趣或令人惊讶的观察结果进行解释。请同时列出重要的代码段并附上解释。只是简单地附上代码不加以解释不会获得学分。

致谢

本实验室得到了 Honghao Zeng 和 Keyi Li 的帮助，他们在 SEED Internet 仿真器的 DNS 模块开发中做出了贡献。本实验的部分经费来自于美国国家科学基金以及雪城大学的赞助。