BGP 探索和攻击实验

1 概述

边界网关协议（BGP）是用于在互联网上的自治系统（AS）之间交换路由和可达性信息的标准外部网关协议。它是互联网的 “粘合剂”，是互联网基础设施的重要组成部分，也是主要的攻击目标之一。如果攻击者能够控制 BGP，则可以断开互联网并重定向流量。

本实验的目标是帮助学生了解 BGP 如何将互联网连接在一起以及互联网是如何连接的。我们构建了一个互联网仿真器，并将使用此仿真器作为实验活动的基础。由于 BGP 的复杂性，我们单独提供了其工作原理的文档，建议学生在进行实验之前阅读该文档。本实验涵盖以下主题：

- BGP 协议的工作方式
- 自治系统（AS）
- BGP 配置、BGP 大型社区
- 路由
- 互联网交换所 (IX)
- BGP 攻击，前缀劫持

支持材料。 BGP 相当复杂，尤其是在实践方面。为了帮助学生完成这个实验，我编写了一章关于 BGP 的内容，已经包含在出版的书中。这一章作为样例章节，供大家免费下载。有关该章节的链接可以从实验网页上找到。如果没有阅读这一章节，完成此实验将会非常困难。

教师注意事项。 任务 1 到 4 是为了帮助学生理解 BGP 的技术细节而设计的，特别适合那些在课程中深入讲解了 BGP 的老师使用。只有任务 5（BGP 攻击）与安全相关，因此如果只希望学生关注 BGP 的安全性方面，则可以跳过任务 1-4，直接让学生做任务 5，因为此任务不依赖于前面的任务。

实验室环境。 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。

2 实验设置与 SEED Internet 模拟器

此实验将在 SEED 互联网仿真器（以下简称仿真器）中进行。我们提供了两种形式的仿真器：Python 代码和容器文件。这些容器文件是 Python 代码生成的，但学生需要从 GitHub 上安装 SEED 仿真器的源代码才可以运行 Python 代码。而容器文件则可以直接使用，无需安装仿真器源代码。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，并将文件提供给学生替换实验提供的文件。

下载仿真器文件。 请从网页上下载 `Labsetup.zip` 文件并解压缩。`output` 文件夹内的文件是从 Python 脚本 mini-internet.py 生成的实际仿真文件（容器文件）。

启动仿真。 我们将直接使用 `output` 文件夹中的容器文件。进到该文件夹，运行以下 Docker 命令来构建并启动容器。我们建议在提供的 SEED Ubuntu 20.04 虚拟机中运行仿真器，但只要安装了 Docker 软件，也可以在一个通用的 Ubuntu 20.04 操作系统上进行操作。读者可以从 此处链接 查找 Docker 手册。

```
$ docker-compose build
$ docker-compose up

// 在SEED VM 中的别名命令（仅在 SEED VM 中可用）
$ dcbuild       # 别名: docker-compose build
$ dcup          # 别名: docker-compose up
$ dcdown        # 别名: docker-compose down
```

2.1 网络图

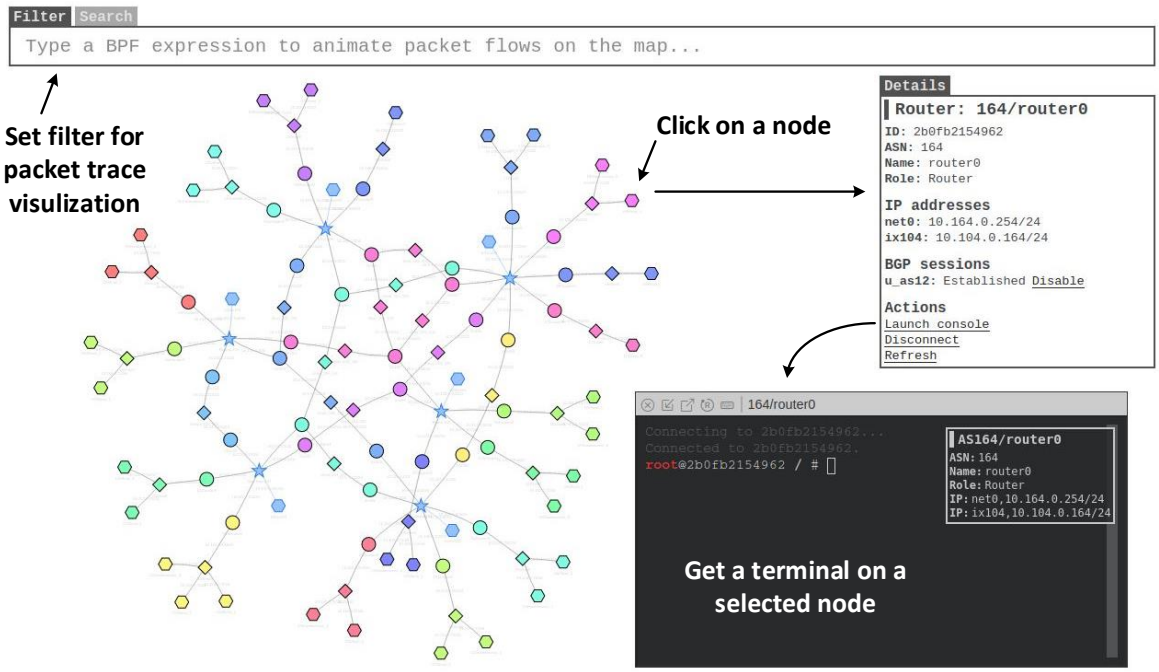

图 1: SEED 互联网仿真器

仿真器内的每台计算机（主机或路由器）都是一个 Docker 容器。用户可以通过使用 Docker 命令来访问这些计算机，例如在容器内部获取 shell。仿真器还附带了一个 Web 应用程序，可以可视化所有主机、路由器和网络。启动仿真器后，可以从以下 URL 访问地图：`http://localhost:8080/map.html`。参见图 1。用户可以通过此地图进行交互操作，例如从容器中获取终端，禁用 BGP 会话（参见图 2）。

此外，还可以设置过滤器来可视化网络流量。过滤器的语法与 tcpdump 的相同，实际上，这些过滤器直接传递给仿真器上所有节点上的 tcpdump 程序。

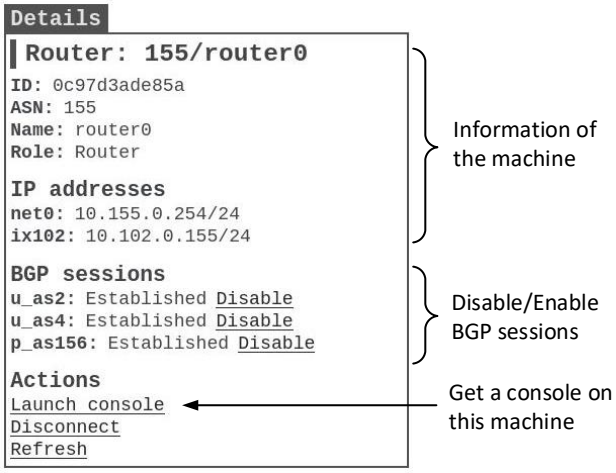

图 2: 与节点交互

2.2 修改 BGP 配置文件

在多个任务中，我们需要修改 BGP 配置文件。可以通过直接在容器内修改配置文件来实现这一点。另一个方法是将文件复制到主机虚拟机，在该主机上进行编辑，然后将其复制回容器。请参见以下示例（假设我们想修改 AS-180 的 BGP 配置文件）：

```
// 查找 AS-180 的 BGP 路由器容器的 ID
$ dockps | grep 180
6bf0bcda8d06  as180h-host_1-10.180.0.72
437874056b15  as180h-webservice_0-10.180.0.71
29676d5034ce  as180r-router0-10.180.0.254     ← 这是 AS-180 的 BGP 路由器

// 将配置文件从容器复制到主机机器
$ docker cp 2967:/etc/bird/bird.conf ./as180_bird.conf

... 编辑该文件 ...

// 将文件复制回容器
$ docker cp ./as180_bird.conf 2967:/etc/bird/bird.conf

// 在容器内重新加载配置
$ docker exec 2967 birdc configure    ← 运行"birdc configure"
BIRD 2.0.7 ready.
Reading configuration from /etc/bird/bird.conf
Reconfigured
```

2.3 在仿真器中使用的约定

为了使用户能够容易地识别仿真实验中的各个节点的角色，我们创建了一套给各种节点分配不同编号时要遵循的约定。这些约定仅适用于仿真实验，在现实世界中并不适用。

- 自治系统号码（ASN）分配：
  - ASN `2 - 9`：大型中转自治系统（例如国家骨干网）。
  - ASN `10 - 19`：较小的中转自治系统。
  - ASN `100 - 149`：互联网交换所 (IX)。
  - ASN `150 - 199`： Stub 自治系统（Stub AS）。
- 网络前缀和 IP 地址：
  - 对于自治系统 N,其第一个内部网络的前缀为 `10.N.0.0/24`,第二个内部网络为 `10.N.1.0/24`，依此类推。
  - 在每个网络中，地址从 `200` 到 `255` 都是路由器的 IP 地址。对于非路由器（主机），其 IP 地址从 `71` 开始。例如，在 AS-155 中，`10.155.0.255` 是一个 BGP 路由器，而 `10.155.0.71` 则是一个主机。

3 任务 1：Stub 自治系统

自治系统（AS）是 BGP 的基本单位。stub AS 是不向其他 AS 提供中转服务的 AS 类型。大多数终端用户都是 stub AS，包括大学、组织和大多数公司。另一种类型的 AS 称为中转 AS。他们为其他 AS 提供中转服务，他们是互联网服务提供商。

在这个任务中，我们将专注于 stub 自治系统，看看它们如何与其他自治系统进行互连。对于这种类型的自治系统，我们可以通过了解其 BGP 配置来获得对 BGP 工作方式的初步了解。学生在开始此任务之前应先阅读章节 1- 7 。

3.1 任务 1.a：理解 AS-155 的 BGP 配置

AS-155 是一个 stub 自治系统,它有一个网络 (`10.155.0.0/24`) 和一个 BGP 路由器(`10.155.0.254`)。请在该路由器容器上获取终端，并研究 `/etc/bird/bird.conf` 中的 BGP 配置，然后完成以下任务。

- **任务 1.a.1：** 从 BGP 配置文件中识别 AS-155 与其他哪个 AS 进行互连。目前可以忽略配置中的过滤部分。以下是配置文件中的一个条目。有关每个条目的解释，请参阅提供的 BGP 教程中的 Section 6 。

```
protocol bgp u_as2 {
    ipv4 {
        table t_bgp;
        import filter {
            bgp_large_community.add(PROVIDER_COMM);
```

```
            bgp_local_pref = 10;
            accept;
        };
        export where bgp_large_community ~ [LOCAL_COMM, CUSTOMER_COMM];
        next hop self;
    };
    local 10.102.0.155 as 155;
    neighbor 10.102.0.2 as 2;
}
```

- **任务 1.a.2：** AS-155 与其他多个 AS 进行互连，因此如果与其中一个 AS 的连接中断，它仍然可以继续连接到互联网上。请设计一个实验来证明这一点。可以从图中（见图 2）或使用 `birdc` 命令（参见以下示例）来启用/禁用 BGP 会话。在您的实验中，请展示当特定的 BGP 会话被禁用/启用时路由表的变化。（您可以使用 `"ip route"` 来查看路由表的内容。）

```
root@0c97d3ade85a / # birdc show protocols
BIRD 2.0.7 ready.
Name       Proto      Table      State  Since         Info
u_as2      BGP        ---        up     14:51:40.447  Established
u_as4      BGP        ---        up     14:51:39.500  Established

root@0c97d3ade85a / # birdc disable u_as2   ← 禁用与 AS-2 的互连
BIRD 2.0.7 ready.
u_as2: disabled

root@0c97d3ade85a / # birdc show protocols
BIRD 2.0.7 ready.
Name       Proto      Table      State  Since         Info
u_as2      BGP        ---        down   16:32:14.883
u_as4      BGP        ---        up     14:51:39.500  Established
```

3.2 任务 1.b：观察 BGP UPDATE 消息

本任务的目的是了解 BGP UPDATE 消息。在 AS-150 的 BGP 路由器上运行 `tcpdump`，使用它来监视 BGP 流量。此命令将把捕获的 BGP 数据包保存到 `/tmp/bgp.pcap` 中。

```
# tcpdump -i any -w /tmp/bgp.pcap "tcp port 179"
```

您的任务是通过 AS-155 的 BGP 路由器做一些事情来触发至少一个 BGP 路由撤销和一个 BGP 路由通告消息。这些 UPDATE 消息会被 `tcpdump` 命令捕获并存储在 `bgp.pcap` 中。使用 `"docker cp"` 命令将此文件复制到主机计算机，然后将其加载到 Wireshark 中。选择一个路由通告消息和一个路由撤销消息，提供对这两个消息的解释。屏幕截图应在实验报告中提供。

3.3 任务 1.c：试验大型社区

当 BGP 路由器向其邻居发送路由时，并非会发送所有已知的路由。哪些路由被发送取决于许多因素，例如邻居的位置、邻居之间的业务关系以及策略。为了帮助 BGP 路由器做出这些决策，需要在每条路由上附加一些额外信息，预定义的路由属性无法包含此类信息。BGP 大型社区正是为此创建的。本任务的目标是在仿真实验中了解如何使用它来反映邻居之间的业务关系。

假设由于某些技术问题，AS-4 与 AS-156 的连接中断了。我们可以通过关闭 AS-4 和 AS-156 之间的 BGP 互连来模拟此情况。由于 AS-4 是 AS-156 唯一的服务提供商，因此这实际上就将 AS-156 与互联网断开了。如果从 AS-156 的某个主机 ping 另一个主机，可以看到以下结果（请勿在 BGP 路由器上运行 ping，在主机上做）：

```
// 在 10.156.0.72 上
# ping 10.155.0.71
PING 10.155.0.71 (10.155.0.71) 56(84) bytes of data.
64 bytes from 10.155.0.71: icmp_seq=1 ttl=62 time=14.6 ms
64 bytes from 10.155.0.71: icmp_seq=2 ttl=62 time=0.363 ms

# ping 10.161.0.71
PING 10.161.0.71 (10.161.0.71) 56(84) bytes of data.
From 10.156.0.254 icmp_seq=1 Destination Net Unreachable
From 10.156.0.254 icmp_seq=2 Destination Net Unreachable
From 10.156.0.254 icmp_seq=3 Destination Net Unreachable
```

我们看到 10.155.0.71 仍然可达，因为它属于 AS-155，AS-155 和 AS-156 还是互连的。然而，10.161.0.71（归属于 AS-161）则无法访问，因为没有人会为 AS-156 路由该包。AS-156 仍然与 AS-155 进行互连，并且 AS-155 直接连接到互联网，那么为什么 AS-156 不能通过 AS-155 访问互联网呢？这是因为是否转发另一个自治系统中的流量取决于双方之间的业务关系。

在 AS-156 和 AS-4 试图解决问题的同时，AS-156 找到了 AS-155，同意支付给 AS-155 一定的费用，暂时通过 AS-155 访问互联网。这需要对 AS-155 的 BGP 路由器进行一些更改，以便 AS-155 可以临时为 AS-156 提供中转服务。请参阅 Section 9 完成此任务，并确保运行以下命令重新加载 BIRD 配置。

```
# birdc configure
BIRD 2.0.7 ready.
Reading configuration from /etc/bird/bird.conf
Reconfigured
```

3.4 任务 1.d：配置 AS-180

AS-180 已包含在仿真器中。它连接到 IX-105 互联网交换所，但不与任何人互连，因此未连接到互联网上。在本任务中，学生们需要完成 AS-180 的 BGP 路由器及其所有相关的 BGP 路由器的配置，以实现以下目标：

- 和 AS-171 进行互连，使得他们可以直接互相通信。

- 和 AS-2 和 AS-3 这两个中转自治系统进行互连，以便它们可以通过这些中转到达其他目的地。

Shell 脚本。 在本任务中，我们需要修改多个 BIRD 配置文件。逐个进入容器进行更改不是太方便，我们可以将所有 BIRD 配置文件从容器复制到主机虚拟机，在该主机上进行编辑，然后将它们复制回容器。我们在 `task1` 文件夹中包含了两个 shell 脚本，以简化此过程：

- `import_bird_conf.sh`: 获取所有所需的 BIRD 配置文件。如果当前目录中已经存在这个配置文件，该文件不会被覆盖。
- `export_bird_conf.sh`: 将 BIRD 配置文件复制到容器，并运行 `"birdc configure"` 命令以重新加载配置。

调试。 如果结果不对，需要进行调试以找出哪里出了问题，特别是找到数据包去了哪里。例如，如果运行了 `ping` 但没有收到回复，可以使用地图客户端的过滤选项可视化流量，找到包去了哪里。过滤选项的语法和 `tcpdump` 的语法相同。我们给出了以下示例。

```
"icmp"                       ← 显示所有 ICMP 流量
"icmp and src 10.180.0.71"   ← 显示来自 10.180.0.71 的 ICMP 流量
"icmp and dst 10.180.0.71"   ← 显示去往 10.180.0.71 的 ICMP 流量
```

实验报告。 在实验报告中，请提供添加到 BIRD 配置文件中的内容，并给出适当的解释。同时，请提供截图（如 traceroute）以证明任务成功完成。

4 任务 2：中转自治系统

如果两个自治系统想要互相连接，它们可以在互联网交换所互连。问题是，两个位于不同位置的自治系统如何彼此互相通讯？它们很难找到一个共同的地方进行互连。为了解决这个问题，需要一种特殊类型的自治系统。

这种自治系统在许多互联网交换点（IX）都有 BGP 路由器，并与其他多个自治系统进行互连。一旦数据包进入其网络，这个自治系统把数据包从一个 IX 运到另一个 IX（通常通过一些内部路由器），最终交给另一个自治系统处理。正是由于这种自治系统为其他自治系统提供了中转服务，互联网上的机器才可以互相通讯。这种特殊的自治系统 b 被称为*中转自治系统* (Transit AS)。

在本任务中，我们将首先了解中转自治系统的运作方式，然后将在我们的互联网仿真实验中配置一个中转自治系统。学生们应在开始此任务之前阅读教程中的 Section 10 。我们在这个任务中选择了 AS-3 中转自治系统。这个 AS 有四个 BGP 路由器，每个路由器都在不同的互联网交换所（IX）。

4.1 任务 2.a：IBGP 的实验

对于此任务，首先需要找到一些经过 AS-3 的流量。我们将从 AS-162 中的一个主机 ping `10.164.0.71`。使用地图客户端程序可以看到，数据包通过 AS-3 中转自治系统转发。如果您的观察结果与此不符，请尝试找到一些其他经过 AS-3 的流量。

现在我们在 AS-3 在 IX-103 的 BGP 路由器上关闭 IBGP 会话（使用地图客户端或从命令行执行此操作，参见以下示例）。

```
# birdc
bird> show protocols
Name       Proto      Table      State  Since         Info
...
ibgp1      BGP        ---        up     20:19:03.800  Established
ibgp2      BGP        ---        up     20:19:11.921  Established
ibgp3      BGP        ---        up     20:20:50.238  Established

bird> disable ibgp3
bird> show protocols ibgp3
Name       Proto      Table      State  Since         Info
ibgp3      BGP        ---        down   20:26:44.526
```

在关闭 IBGP 之前，请先在 BGP 路由器上显示路由表（使用 "ip route"）。比较关闭 IBGP 前后的结果，并解释您的观察结果。

4.2 任务 2.b：IGP 试验

在这个任务中，我们将使用同一个 BGP 路由器。我们将会关闭 OSPF 路由协议，并观察其对路由的影响。有几种方法可以关闭 OSPF。一种方法是在 birdc 内进行操作：

```
# birdc
birdc> show protocols
...
ospf1      OSPF       t_ospf     up     19:49:43.343  Running
...

birdc> disable ospf1
birdc> show protocols ospf1
ospf1      OSPF       t_ospf     down   19:57:37.187
```

在关闭和打开 OSPF 前后，请在 BGP 路由器上显示路由表（使用 "ip route"）。比较结果。基于观察，解释为什么 IGP 对于中转自治系统是至关重要的。

4.3 任务 2.c：配置 AS-5

中转自治系统 AS-5 已包含在仿真实验中。它连接到三个互联网交换所：IX-101、IX-103 和 IX-105，但不与其他任何 AS 进行互连。其拓扑结构可以参见图 3。在本任务中学生需要完成以下工作：

- 在仿真实验中，AS-5 的 IBGP 会话已经建立。请以 IX-101 上 AS-5 的 BGP 路由器为例，解释其 IBGP 配置的意义。

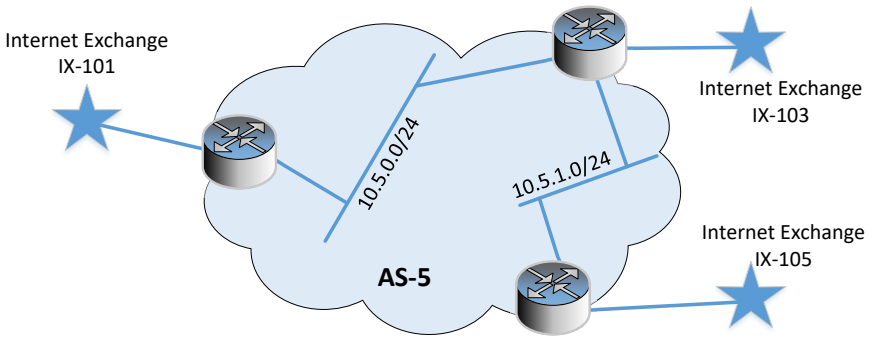

图 3: AS-5 的网络图

- AS-5 将为 AS-153（IX-101）、AS-160（IX-103）和 AS-171（IX-105）提供中转服务。请相应地配置它们的 EBGP 互连。学生可以从其他中转 AS 中学习如何配置其互连。
- AS-5 和 AS-3 都是中转 AS，它们决定在 Miami 的 IX-103 上进行互连。由于两者规模大致相当，因此双方都从这种互连中获益，所以它们决定彼此之间的关系应为对等关系，而非提供方与客户的关系。他们之间不需要付费。

在这个任务中，我们可能需要修改几个 BIRD 配置文件。就像在任务 1 中一样，我们在 `task2` 文件夹中创建了两个 shell 脚本，可以用于从容器中自动下载/上传 BIRD 配置文件。

在实验报告中，请提供添加到 BIRD 配置文件中的内容，并给出适当的解释。同时，请提供截图（如 traceroute）以证明任务成功完成。

5 任务 3：路径选择

BGP 路由器通常会接收到通往同一网络的多条路径。这些路径都将被保留，但 BGP 将通过最佳路径选择算法来选择一条作为当前最优路径。这条路径将被宣布给邻居，同时也被交给内核路由，本机上的路由将取决于这个选中的路径。当这个最佳路径被撤销时，BGP 路由器会重新通过最佳路径选择算法找到新的当前最优路径。

BIRD 实现的最佳路径选择算法中优先级最高的两个标准如下：（1）优先选择具有最高本地偏好属性的路径（Local Preference），（2）优先选择 AS 路径最短的路径。在本任务中，我们将对这两个标准做些实验，观察它们如何影响路径选择。学生们在开始此任务之前应先阅读教程中的 Section 8 。

任务 3.a 使用 `"ip route"` 命令可以列出内核路由表中的所有条目。使用以下命令可以显示所有 BGP 路径（输出很长，因此最好将其保存到一个文件中）：

```
"birdc show route all > all-routes"
```

请转到 AS-150 的 BGP 路由器上显示所有 BGP 路径。找到具有多个路径的网络前缀，并解释这些路径之间的差异。将这些路径与内核路由表中的相应条目进行比较，然后指出哪条 BGP 路径被选中成最佳路径，并解释为何选择此路径。

任务 3.b AS-150 同时与 AS-2 和 AS-3 进行互连，两者都向 AS-150 提供互联网服务。然而，由于（假设）到 AS-2 的链路比到 AS-3 的链路慢得多，AS-150 希望仅将 AS-2 用作备份的路径，平常只用 AS-3，除非 AS-3 这条路断了。请修改 AS-150 的 BGP 配置以实现此目标。

6 任务 4：IP 任播（anycast）

IP 任播指的是单个 IP 地址由多个机器（通常位于不同位置）共享，当我们向这个 IP 地址发送一个数据包时，其中一个机器会收到此数据包。具体哪个机器会收到很难预测，因为这取决于路由。IP 任播是 BGP 主持的。它的一个众所周知的应用是 DNS，根服务器 A-M 只有 13 个 IP 地址，但拥有这些 IP 地址的机器远远超过这个数目。

在仿真器的地图的搜索框中输入 `190` ，您可以发现 AS-190 有两个网络，但它们是断开的。一个网络连接到 IX-100，另一个连接到 IX-105。仔细查看这两个网络会发现它们拥有相同的网络前缀 `10.190.0.0/24`。这两个网络上的唯一主机具有相同的 IP 地址 `10.190.0.100`。

请在仿真实验中找到两个不同的主机，当您从这些主机 ping `10.190.0.100` 时，目的地将会不同（尽管目标 IP 地址相同）。设置过滤器以显示 `icmp` 流量，然后查看路径上的 BGP 路由器，并解释为何数据包到达不同的目的地。学生应在开始此任务之前阅读教程中的 Section 11 。

7 任务 5：BGP 前缀攻击

BGP 前缀劫持是一种典型的 BGP 攻击。在这种攻击中，攻击者的 BGP 路由器会对外宣称它是某些 IP 前缀的拥有者，但其实它并不是，这些 IP 前缀是属于受害者的。这将导致前往此 IP 地址的流量被路由到攻击者那里，被攻击者拦截或修改。学生们应在开始此任务之前阅读教程中的 Section 12 。

7.1 任务 5.a. 使用 AS-161 的 BGP 路由器执行前缀劫持攻击

在此任务中，我们将使用 AS-161 中的 BGP 路由器发起前缀劫持攻击。我们的目标是劫持属于 AS-154 的 IP 前缀。如果攻击成功，所有前往 AS-154 的数据包都会被路由到 AS-161，在那里它们会被丢弃。看上去，去往 AS-154 的数据包都进了 “黑洞”。

在 BIRD 中，BGP 路由器所宣布的前缀可以来自不同的来源：这些前缀可能来自 `direct` 协议，即从附着到 BGP 路由器的实际网络接口获取。它们也可以来自 `static` 协议，其中包含预定义的路由。BGP 路由器要宣布一个它不拥有的前缀的最简单方式是使用 `static` 协议。以下示例创建了一个静态路由到 `10.130.0.0/16` 前缀，有关详细解释，请参阅教程中的 Section 3 。应注意的是，在该条目的过滤部分，我们需将其添加到 `LOCAL_COMM` 社区中，否则，BGP 路由器将不会向外出口此路由。

```
protocol static {
    ipv4 {
        table t_bgp;
    };
    route 10.130.0.0/16 blackhole { bgp_large_community.add(LOCAL_COMM); };
}
```

7.2 任务 5.b. AS-154 的反击

AS-154 具有检测系统，攻击发起后，它立即检测到了攻击。它尝试打电话联系 AS-161 的上游服务提供商 AS-3，请他们阻止该攻击。不幸的是，AS-3 没有人接电话。由于服务中断导致的损失严重，AS-154 决定反击，希望夺回自己的网络前缀，而不是依赖于 AS-3。请重新配置 AS-154 的 BGP 路由器以成功实现此目标。

7.3 任务 5.c. 在 AS-3 处解决问题

最终，AS-3 联系上了。在不明了这是由 AS-161 的误配引起还是故意攻击的情况下，AS-3 决定不断开和 AS-161 的 BGP 连接，这样 AS-161 的用户仍可访问互联网（AS-3 是 AS-161 的唯一服务提供商）。然而，AS-3 必须停止错误路径的传播。这可以通过从其自己的公告中删除这些虚假的路径。更具体地说，AS-3 可以在其过滤器代码中添加一些内容，这样就能屏蔽掉虚假的路径。请参阅 BGP 教程中的 Section 12 以了解如何编写过滤器。

8 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

本实验得到了雪城大学电气工程与计算机科学系研究生曾鸿浩的帮助。