防火墙规避实验：使用 VPN 绕过防火墙

1 概述

组织、互联网服务提供商（ISPs）和国家通常会阻止其内部用户访问某些外部网站。这被称为出口过滤 (egress filtering)。例如，为了防止工作时间的分心，许多公司会在他们的出口防火墙上设置规则以屏蔽社交网络站点，这样员工就无法从公司的网络中访问这些网站。出于政治原因，许多国家在 ISP 层面设置了出口过滤，屏蔽其人民对特定外国网站的访问。不幸的是，这些防火墙可以很容易地被绕过，并且帮助用户绕过防火墙的服务/产品在网上广泛可得。最常用的绕过出口防火墙的技术是虚拟私人网络（VPN）。特别地，这项技术被深受出口过滤影响的智能手机用户广泛使用；有许多适用于 Android、iOS 及其他平台的 VPN 应用程序可以帮助用户绕过出口防火墙。

本实验的学习目标是让学生看到 VPN 是如何工作的，以及 VPN 如何帮助用户绕过出口防火墙。我们将在本次实验中实现一个非常简单的 VPN，并利用它来绕过防火墙。典型的 VPN 依赖于两种技术：IP 隧道和加密。隧道技术是帮助绕过防火墙的最关键的一项技术；而加密技术则是为了保护通过 VPN 隧道传输的内容。为简化问题，我们仅关注于建立隧道的过程，并不对隧道内传输的流量进行加密处理。我们有一个独立的 VPN 实验室涵盖隧道和加密两方面内容。如果读者感兴趣的话，可以完成我们的 VPN 实验以学习如何构建完整的 VPN 系统。在本次实验中，我们将仅关注使用 VPN 隧道绕过防火墙的方法。本实验涵盖以下主题：

- 防火墙
- VPN

阅读材料和视频。 有关防火墙、防火墙规避技术以及 VPN 的详细内容可以在以下几个位置找到

- SEED Book 中的第 17 章和第 19 章，*Computer & Internet Security: A Hands-on Approach*, 3rd Edition, by Wenliang Du. 详情请见 `https://www.handsonsecurity.net`.
- SEED Lecture 中的第 8 节和第 9 节，*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。

2 实验任务

2.1 任务 1：虚拟机设置

我们需要两台机器，一台在防火墙内，另一台在防火墙外。目标是帮助位于防火墙内的那台机器访问被防火墙屏蔽的外部网站。我们使用两个虚拟机 VM1 和 VM2 来模拟这两台机器。VM1 与 VM2 依赖于路由器通过互联网连接。这个设置可能需要超过两台虚拟机。为简化起见，我们将用一个局域网（LAN）来仿真互联网连接。实际上，我们可以简单地使用 `NAT Network` 适配器将 VM1 和 VM2 连接到一个局域网中。图 1展示了实验环境设置。

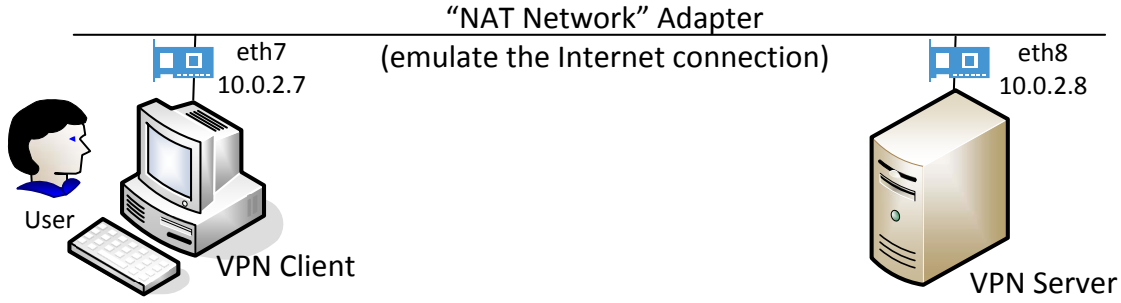

图 1: 实验室环境设置

2.2 任务 2：设置防火墙

在本任务中，您将在 VM1 上设置一个防火墙以阻止对特定目标网站的访问。请确保目标网站的 IP 地址是固定的或在一个固定范围内；否则，你可能无法完全屏蔽该站点。有关如何屏蔽网站的具体操作，请参阅防火墙实验 Firewall lab。

在现实世界中，防火墙应该运行在单独的一台机器上，而不是在 VM1 上。为减少实验中使用的虚拟机数量，我们把防火墙放在了 VM1 上。在 VM1 上设置防火墙需要超级用户权限。同样，设置 VPN 隧道也需要超级用户权限。有人可能会立即说，如果我们已经有超级用户的权限，为什么不能直接关闭 VM1 上的防火墙。这是一个很好的论点，但是请注意，我们将防火墙放在 VM1 上仅仅是因为我们不想在实验环境中创建另一台虚拟机。因此，尽管您在 VM1 上有超级用户权限，但您不被允许使用该权限重新配置防火墙。您必须使用 VPN 来绕过它。

与将防火墙放置在外置机器相比，在 VM1 上设置防火墙确实有一些小问题需要处理。当我们设置防火墙以阻止数据包时，我们需要确保不会阻止到达用于 VPN 的虚拟接口的数据包，否则我们的 VPN 也无法接收到这些数据包。因此我们不能在路由之前或在虚拟接口上设置防火墙规则。我们只需要在 VM1 的实际网络接口上设置规则，这样就不会影响到流向虚拟接口的数据包了。以下命令可以阻止所有流量到达 `93.184.216.0/24` 网络 (`example.com`)：

```
$ sudo iptables -A OUTPUT -o enp0s3 -d 93.184.216.0/24 -j DROP
```

请确定一个您想要屏蔽的网站，设置防火墙，并验证您的防火墙工作正常且目标 IP 地址不再可访问。在实验报告中提供屏幕截图。

2.3 任务 3：使用 VPN 绕过防火墙

使用 VPN 来绕过防火墙的想法如图 2所示。我们在 VM1（VPN 客户端虚拟机）和 VM2（VPN 服务器虚拟机）之间建立一个 VPN 隧道。当 VM1 上的用户尝试访问被屏蔽的网站时，数据包不会直接通过其网络适配器传输，因为那边会被拦截阻止。取而代之，来自 VM1 的数据包将被路由到 VPN 隧道，并到达 VM2。一旦到达那里，VM2 将把这些数据包转发至最终目的地。当回复数据包返回时，它会回到 VM2，然后由 VM2 重定向这些数据包通过隧道，最后把数据包传回给 VM1。这就是如何使用 VPN 帮助 VM1 绕过防火墙的。

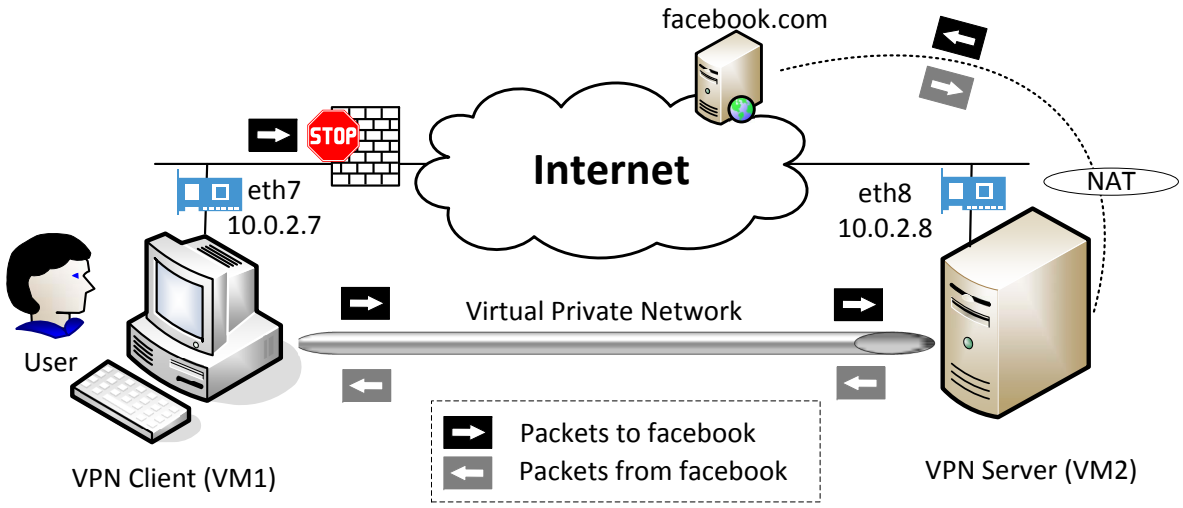

图 2: 使用 VPN 绕过防火墙

我们创建了一个样本的 VPN 程序，包括客户端程序 (`vpnclient`) 和服务器程序 (`vpnserver`)，它们都可以从该实验室网站下载。这个简单的 VPN 程序仅在客户端和服务器之间建立一个隧道；它不对隧道中的流量进行加密。此程序具体细节的解释可以在 SEED 书籍的 VPN 章节中找到。

`vpnclient` 和 `vpnserver` 程序是 VPN 隧道的两端。它们通过图 3中所示的套接字，使用 TCP 或 UDP 进行通信。在我们的示例代码中，为简化起见我们选择使用 UDP。客户端和服务器之间虚线表示 VPN 隧道路径。VPN 客户端和服务器程序通过 TUN 接口连接到主机系统，并通过它完成以下两项工作：（1）获取来自主机系统的 IP 包以便可以通过隧道传输；（2）从隧道获取 IP 包并将其转发给主机系统，该系统会将数据包发送到最终的目的地。下面的步骤描述了如何使用 `vpnclient` 和 `vpnserver` 程序创建一个 VPN 隧道。

步骤 1：运行 VPN 服务器 我们首先在 Server VM 上运行 VPN 服务器程序 `vpnserver`。运行此程序后，系统中会显示一个新的虚拟 TUN 网络接口（我们可以使用 `"ifconfig -a"` 命令查看；大多数情况下接口名称为 `tun0`，但也可以是一个数字 `tunX`）。此时该新接口尚未配置好，我们需要通过分配

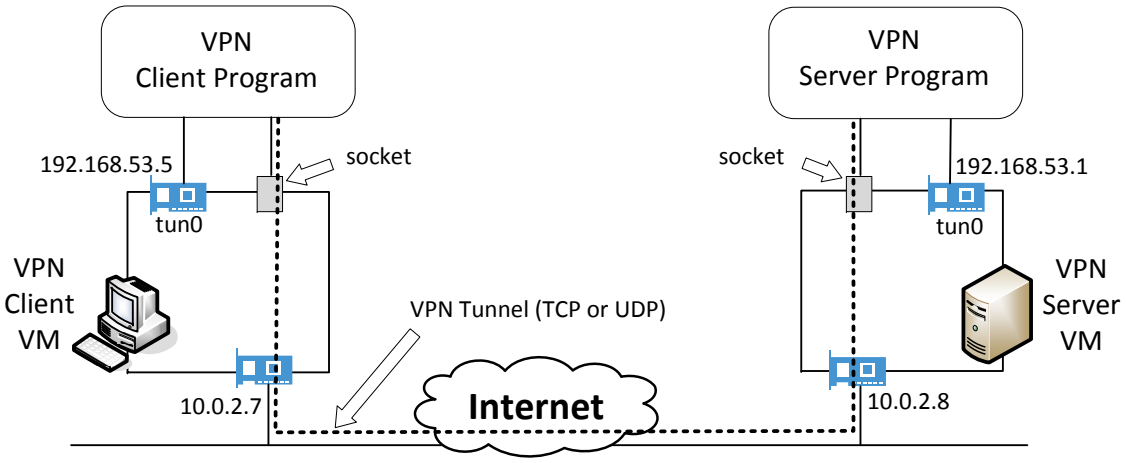

图 3: 客户端和服务器端程序

IP 地址来对其进行配置。我们通常为这个接口指定 192.168.53.1 的 IP 地址，不过您也可以使用其他 IP 地址。

运行以下命令。第一个命令将启动服务器程序，第二个命令则会向 tun0 接口分配一个 IP 地址并激活它。需要特别注意的是，第一个命令将会阻塞且等待连接，因此我们需要在另一个窗口中执行第二个命令。

```
$ sudo ./vpnserver

Run the following command in another window:
$ sudo ifconfig tun0 192.168.53.1/24 up
```

如果没有特别配置，计算机通常仅作为主机工作而不会充当网关。VPN 服务器需要将数据包转发到其他目的地，因此它需要像网关一样运行。为了使计算机像网关那样行为我们需要启用 IP 转发。可以通过以下命令启用 IP 转发：

```
$ sudo sysctl net.ipv4.ip_forward=1
```

步骤 2：运行 VPN 客户端 我们现在在 Client VM 上运行 VPN 客户端程序。我们在此机器上执行如下命令（第一个命令将连接到运行在 10.0.2.8 的 VPN 服务器程序）。此命令也将阻塞，因此我们需要在一个不同的窗口中配置由 VPN 客户端创建的 tun0 接口，并给 tun0 接口分配 IP 地址 192.168.53.5（您可以选择其他 IP 地址）。

```
On VPN Client VM:
$ sudo ./vpnclient 10.0.2.8

Run the following command in another window:
$ sudo ifconfig tun0 192.168.53.5/24 up
```

步骤 3：在客户端和服务器虚拟机上设置路由。 经过上述两个步骤，隧道将建立起来。在使用该隧道之前，我们需要在这两台机器（客户端和服务端）上设置路由路径，以便通过该隧道传送期望的流量。我们可以使用 `route` 命令添加一个路由条目。以下示例演示如何将发往 `10.20.30.0/24` 的数据包导向接口 `eth0`。

```
$ sudo route add -net 10.20.30.0/24 eth0
```

为绕过客户机虚拟机中的防火墙，您需要设置相应的路由条目。这样被阻止网站的流量会被导向到 VPN 中。您需要考虑添加哪些路由条目来绕过防火墙。

步骤 4：在服务端虚拟机上进行 NAT 的设定。 当最终目的地向用户发送数据包时，该数据包将首先发送到 VPN 服务器（请思考原因，并将其答案写入报告中）。返回的数据包会先到达 VPN 服务器的 NAT 适配器（因为所有从服务端虚拟机发出的数据包的源 IP 地址都会被更改成 NAT 的外部 IP 地址，也就是在我们的设置中基本等同于主机计算机的 IP 地址）。通常情况下，NAT 会将目标 IP 地址替换为原始数据包中的 IP 地址（即我们的情况下的 `192.168.53.5`），并将它回传给拥有该 IP 地址的人。不幸的是，在这里我们遇到了问题。

在 NAT 发送出数据包之前，需要知道拥有 `192.168.53.5` 的机器的 MAC 地址，因此会发出一个 ARP 请求。我们的私有网络是虚拟的，并且这个 IP 地址属于 VPN 客户端上的 `tun0` 接口。所以 `192.168.53.5` 不会接收到 ARP 请求（即使它收到了也没有用）。NAT 将丢弃该数据包，因为接收者不存在。

实际的接收者应该是服务器虚拟机，尽管它并不拥有 `192.168.53.5`。如果可以配置 NAT 作为网关，我们可以要求 NAT 将发往 `192.168.53.5` 的数据包路由到 VPN 服务器，最终通过隧道传递给 VPN 客户端。然而，在 VirtualBox 中我们还没有找到如何配置 NAT 作为网关的方法。不过，我们找到了两种可能的方法来应对这个问题。其中一个想法是让 NAT “相信” `192.168.53.5` 的 MAC 地址是服务器虚拟机的 MAC 地址，这样数据包就会通过 NAT 被发送到服务器虚拟机。我们可以使用在 NAT 上进行 ARP 缓存欺骗的方法实现这一点，也就是提前告诉 NAT 关于 `192.168.53.5` 的 MAC 地址。

解决此问题的一种更好方法是在服务端虚拟机创建另一个 NAT。这样所有从服务端虚拟机发出的数据包的源 IP 都会是这个虚拟机自身的 IP。为了连接到互联网，这些数据包会经过由 VirtualBox 提供的第二个 NAT。由于源 IP 是服务端虚拟机，因此第二个 NAT 不会有任何问题地将来自 Internet 的返回数据包转发回服务端虚拟机。使用此解决方案，我们不再需要使用 ARP 缓存欺骗来 “愚弄” NAT 了。以下命令可以在服务器虚拟机上启用 NAT （在您的情况下，`NAT Network` 适配器的名字可能不是 `enp0s3`; 您只需要找到您虚拟机上的真实名字即可）：

```
$ sudo iptables -t nat -A POSTROUTING -j MASQUERADE -o enp0s3
```

演示。 如果您正确完成了上述步骤，你应该能够绕过防火墙。你应该展示您可以通过您的客户端机器上的 VPN 连接来访问被封锁的网页。你的解决方案应该不仅能处理网路流量，还应适用于所有其他类型的流量。例如，如果被屏蔽机器上运行了一个 `telnet` 服务器，您应该可以从客户端虚拟机 `telnet` 到这个被屏蔽的服务器。

在实验报告中，你需要提供证据来证明您的流量确实是通过 VPN 隧道传输的，而不是通过某些 “旁门” 传输的。最好的方法是使用 Wireshark 捕获网络流量，并用截取的数据包路径图示描述路径。

如果没有这种证据，我们无法知道你的成功是因为防火墙配置错误（即目标网站本来就没有被屏蔽）还是因为您的 VPN。

3 提交要求

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。