# Network Security

**Assignment 4C, Wednesday, May 16, 2018, version 1.0**

**Handing in your answers:** Submission via Blackboard (http://blackboard.ru.nl)

**Deadline:** Wednesday, June 6, 23:59:59 (midnight)

**Teaching assistants.** Please email *all* of us if you have a question.


- Pol Van Aubel <pol.vanaubel@cs.ru.nl>
- Daan Sprenkels <dsprenkels@science.ru.nl>
- Wouter Kuhnen <w.j.a.kuhnen@student.ru.nl>


The next lecture is in two-and-a-half weeks time. Therefore, assignment 4 is rather large. We split this assignment in three parts so that the grading is distributed somewhat. However, this does not mean that each part has an equal amount of work. When done with one part, move immediately on to the next. Remember that 6 hours per week is the intended time investment for a 3EC course. Don't spend more than 6 hours on part 4a, and make sure that you have 6 hours left for part 4c.

In this assignment you will be using the following tools:

- wireguard: https://wireguard.com
- openvpn: https://openvpn.net/index.php/open-source/documentation.html

Again, do not compile these programs from source, but install them using your distribution's package manager.

Depending on your Linux distribution, Wireguard may not (yet) be available in its official repositories. You may need to add an external repository, then use your Linux distribution's package manager to install Wireguard. The procedure for several different distributions is described on https://www.wireguard.com/install/. If you are using Kali linux, you must follow the instructions for Debian.

Please turn in all your work in plain text files (program source files are also plain text). If you prefer a document with formatting for whatever reason, like including images, use the PDF format to turn in your work (most editors allow you to export to PDF). Note that it's okay to include images separately and then refer to them from within the text files.

Many commands in this assignment need to be run with root rights. This is denoted by a prefix `#`. When a command should be run without root rights, it will be prefixed with `$`. Do not include the prefix when typing the command.

This assignment is about using Virtual Private Networks, or VPNs. Even though you have not yet been given the lecture on VPNs, you should be able to complete this exercise by using the available documentation of the software used. After all, the lecture is not a tutorial on how to use particular software. As usual, if you get stuck, or don't understand some concept that is essential to continue, do not hesitate to send us an e-mail.

1. CNCZ provides an OpenVPN-based Science VPN, which you may find useful at some point. Instructions for this are at http://wiki.science.ru.nl/cncz/index.php?title=Vpn&setlang=en#OpenVPN_.5Bvoor.5D.5Bfor.5D_Linux_.26_MacOS.

   (a) See if you can get this to work with your Science account. If you are unable to do this within 30 minutes, skip to the next exercise and come back if you have time left over.

   (b) Look up in the OpenVPN man page (`man openvpn`) what each line of the configuration file means. For easy searching, append "--" to the first word on the line. So searching for "dev" becomes "--dev".

   (c) Perform traceroutes (`traceroute`) to blackboard.ru.nl, www.google.com, www.cs.ru.nl, and to the VPN server itself, with and without the VPN running. Paste the commands you used and their output to a file. Look at the routing table (`ip route show`), and paste it as well.

   (d) Explain the differences between the traceroutes, paying special attention to the one to the VPN server itself.

   (e) Explain why it is not straightforward to run other services on the OpenVPN server and contact them via the VPN tunnel. Can you think of a solution for this problem?

For the next exercises you must set up a Wireguard network between two machines. These can be physical machines, e.g. your and your lab partner's laptops, both booting some form of Linux, e.g. the Kali USB stick. Note you can make such a stick yourself using the compressed image available at `https://www.cs.ru.nl/~paubel/netsec/2018/netsec-kali-usb-image-20180418.img.gz` (with credentials root:toor). You can extract this to any USB stick that is at least 8004829184 bytes in size. This does destroy any partitions and data already present on the stick.

Note that you will not be able to reach each other's machines through eduroam, but a direct link using an ethernet cable or an ad-hoc WiFi network usually works. The netsec-wep and netsec-wpa networks should also be usable for this purpose.

You can also use virtual machines. Virtualbox and KVM/QEMU are decent options in this regard. The bootable USB iso image linked above should also be directly bootable as a virtual machine disk.

When using virtual machines, the easiest option is setting up two separate VMs and using VM-to-VM networking. You can do this on a Windows host. On the bootable Kali USB sticks, there is not enough storage to run more than one VM, and if your machine does not have sufficient RAM you may find your system freezing. Setting up virtual machines with Virtualbox is covered in plenty of tutorials so we will not cover that here.

Start by reading the frontpage of the Wireguard project at `https://www.wireguard.com/`, paying particular attention to the Conceptual Overview, which includes Cryptokey Routing.

Unless stated otherwise, you are only allowed to use the `wg` command and network configuration commands such as `ip route` and `ip address`. In particular, you are *not* allowed to use `wg-quick`, a quick-setup script provided by Wireguard.

You will have to generate keys to exchange between Wireguard peers. Don't worry, they're rather short and should be easy to transfer even by manually typing them.

Bear in mind that due to the nature of Cryptokey Routing, if AllowedIPs stanzas for multiple peers on a single Wireguard interface overlap, they behave the same as routes: more specific routes overrule more generic routes.

2. Create a folder called `exercise2` to hold your answers and configuration files.

   You *will* have to manually set an IP address and network mask on the interface. You may also need to add a route manually.

   (a) Using the Wireguard documentation, the minimum setup you should get working is a VPN allowing communication between the two machines. Note that neither machine needs to tunnel all its network traffic over the VPN, it only needs to be able to reach the other endpoint through the VPN.

   Try to make the AllowedIPs configuration as restricted as possible.

   Make a shell script of the commands you use, and include configuration files for both endpoints.

   (b) perform a set of short packet captures on both ends of the connection, while doing a ping from one VPN endpoint to the other endpoint and vice versa. The packet captures on each end should be done on two interfaces: one capture on the wg-interface created for the VPN, and another capture on the network interface that is actually carrying the VPN-tunneled traffic (either your normal network interface, or a virtual interface created by e.g. virtualbox). So there should be four captures in total. Include these captures, and name them along the lines of `endpointA-wg.cap` and `endpointB-wlan0.cap`. Also include the commands and (partial) output of the `ping` command and other commands you used during the capture in a file called `capture-commands`.

3. Create a folder called `exercise3` to hold your answers and configuration files.

   Using the Wireguard documentation and the previous exercise's answers, you will now set up the VPN so that one endpoint uses the VPN for all its network traffic, routed by the other endpoint. This is a fairly common usage scenario, and is what is usually meant by "using a VPN service".

   We assume you will accomplish this by using routing tables akin to the one we analyzed in assignment 4B. Again, you are only allowed to use the `wg` command and network configuration commands such as `ip route` and `ip address`. In particular, you are *not* allowed to use `wg-quick`, a quick-setup script provided by Wireguard.

   You *will* have to manually set an IP address and network mask on the interface. You will also need to add routes manually, as in assignment 4B. You will also need to use iptables to NAT the traffic beyond the VPN server, as in assignment 4A.

   (a) Set up Wireguard as a VPN client/server setup, so that the client uses the VPN for all its network traffic.

   Try to make the AllowedIPs configuration for the server as restricted as possible. For the client, the AllowedIPs configuration must allow all IPs on the wireguard tunnel.

   Make a shell script of the commands you use, and include configuration files for both endpoints.

   (b) Perform the same kind of packet capture as in the previous exercise, however, this time the endpoint functioning as the client should ping some host on the internet (e.g. www.google.com) instead of the VPN server. You can use `traceroute` or `mtr` to figure out whether traffic is actually going through the VPN or whether it's taking the normal route to the internet. Include the commands and output of the `ping` and `traceroute` commands you used during the capture in a file called `capture-commands`.

   If you have network issues during this exercise, one thing to do would be to look at your routing table (`ip r show`) and see if you can figure out why traffic is or is not going through the VPN. Of course, send an e-mail if you're stuck.

4. This exercise is *optional*. Create a folder called `exercise4` to hold your answers.

   Wireguard provides a script called `wg-quick` to assist in common setups. It combines two less well-known aspects of the Linux networking stack to make managing Wireguard connections easier, and less likely to leak data outside the VPN. These are firewall marks (fwmark) and rule-based routing, which uses multiple routing tables for different types of traffic.

   Using the documentation of Wireguard and `wg-quick`, try to explain in your own words how this is accomplished. It might help to set up a `wg-quick` connection and inspect your iptables configuration (`iptables-save`, and ip rules configuration (`ip rule list`, `ip route show table <tablename>`).

   Obviously, the restriction on not using `wg-quick` does not apply to this exercise.

5. This exercise is *optional*. Create a folder called `exercise5` to hold your answers.

   Another solution to the network routing problem comes in the form of using network namespaces. In this solution, a Wireguard interface is created and moved to a dedicated network namespace. Then, any program which is run in this namespace can only use that Wireguard interface to perform network communication. Try using network namespaces to set up a client/server Wireguard VPN akin to exercise 3. Helpful documentation can be found at https://www.wireguard.com/netns/.

   Note that in a client/server setup akin to exercise 3, only the client makes use of the namespaces. After all, the server is intended to route between Wireguard and the physical interfaces. However, it is entirely possible to use namespaces on both ends of a wireguard connection, and programs running on either side in their respective namespace would then only ever be able to communicate with each other, over Wireguard.

6. Place the files and directories `exercise1`, `exercise2`, `exercise3`, `exercise4`, and `exercise5`, and all their contents, in a folder called `netsec-assignment4c-STUDENTNUMBER1-STUDENTNUMBER2`. Replace `STUDENTNUMBER1` and `STUDENTNUMBER2` by your respective student numbers, and accomodate for extra / fewer student numbers. Make a `tar.gz` archive of the whole folder and submit this archive in Blackboard.