AI VPN

Release 2021

Civilsphere Project

Jul 11, 2023

AI VPN:

The AI VPN goal is to better protect the privacy of civil society by researching and developing a locally and easy to implement VPN that checks the traffic of devices with AI-based detection to automatically block threats and stop dangerous privacy leaks. The detection of malicious threats, attacks, infections and private leaked data is implemented using novel free software AI technology. The AI VPN was created by the Civilsphere Project at the Stratosphere Laboratory of the Czech Technical University in Prague.

CHAPTER

ONE

CONTENTS

1.1 Motivation

The Civilsphere Project was born in 2018 the Stratosphere Laboratory of the Czech Technical University in Prague. In Civilsphere we believe that NGOs' work, as well the work of journalists, activists, and human rights defenders is a critical asset for our society and we need to protect them. It is their critical work that makes them a highly valuable political target for a wide variety of powerful actors. They receive a continuous flow of attacks and technical abuse the jeopardizes the liberty and free expression in many countries. At Civilsphere we took a step forward to help them by providing free, advanced digital protection against state actors and others.

The AI VPN was designed to help people at risk obtain a quick and real time security assessment of their devices' networkt traffic to identify if they are compromised or at risk. The integration with the Stratosphere Linux IPS provides state-of-the-art machine learning algorithms to detect malware infections.

This project was created in the Artificial Intelligence Centre of the Czech Technical University in Prague in 2021. This project was initially funded by the NL NET foundation.

1.2 Installation

This section explains how to install the components needed to run the AI VPN. Once the code is downloaded, please refer to the next section: Configuration.

1.2.1 Installing Docker & Docker Compose

The AI VPN follows a micro-services design based on Docker and uses Docker Compose to manage the multi-container application. The first step is therefore to install Docker and Docker Compose in the host machine.

To install Docker we recommend following the installation steps instructed in: https://docs.docker.com/engine/install/

To install Docker Compose we recommend following the installation steps as instructed in: https://docs.docker.com/compose/install/

After the installation is successful, check the services are installed:

```
$ sudo docker --version
$ sudo docker-compose --version
```

The AI VPN was developed for:

- Docker version 20.10.5
- docker-compose version 1.25.0

1.2.2 Installing the AI VPN from Source

Download the source code of the AI VPN from GitHub:

```
$ git clone https://github.com/stratosphereips/AIVPN.git
$ cd AIVPN/
```

The AI VPN is a multi-container application. The modules are managed by Docker Compose using the docker-compose.yml file. Each module plays a specific role in the application. The current supported modules are:

- mod_redis: module that uses Redis as the core database for data storage.
- mod_manager: module responsible for the core functionality of the AI VPN.
- mod_comm_recv: module capable of receiving VPN requests from users.
- mod_comm_send: module responsible for sending data and files to users.
- mod_slips: module that runs the Stratosphere IPS for threat detection.
- mod_report: module responsible for reporting the threats found to users.
- mod_openvpn: module provides the OpenVPN service.
- mod_wireguard: module provides the WireGuard VPN service.
- mod_novpn: module provides a special configuration of the OpenVPN service without encryption. This special configuration is to be used only when in places where the use of encryption or VPNs in particular are illegal or can endanger users' life.
- mod_pihole: module provides the AI VPN with protection against trackers and other malicious domains through DNS blocking capabilities.

The next section will cover the configurations needed to run the AI VPN.

1.3 Configuration

This section explains how to create the configuration file that the AI VPN needs to run. Once the configuration is finished, please refer to the next section: Deployment.

1.3.1 Creating the configuration file

The configuration file is used by all the modules. The current configuration has five different sections:

- REDIS: this section contains the address and channel names used by the modules to communicate with each other using a pub/sub scheme.
- LOGS: this section contains the files and directories where each module will store their log files. Note: if the root directory (/logs) is changed, the docker-compose.yml file will also need to be updated to reflect that change.
- STORAGE: this configuration specifies where the user data will be stored, including packet captures, network logs, and incident reports.
- IMAP: this section contains the credentials for the email address to be used to receive automated email VPN requests and send back the VPN profiles for users to connect. Note: we recommend to use a dedicated email account and not your personal account to run this service.
- TELEGRAM: this section contains the credentials for the telegram bot that will receive VPN requests. The configuration also includes the start and waiting messages that will be sent back to the users. See reference guide at: <telegrambot.rst>

- OPENVPN: this section gives the OpenVPN module the basic information needed to run the VPN service.
- WIREGUARD: this section gives the WireGuard VPN module the basic information needed to run the VPN service.
- NOVPN: this section gives the unencrypted OpenVPN module the basic information needed to run the VPN service.
- AIVPN: this section provides application level configurations, including when profiles expire, maximum profiles per account, etc.

Setting up the Configuration File

The AI VPN includes an example configuration file. Make a copy of the example configuration into a new file in the same folder:

```
$ cd AIVPN/
$ cp config/config.ini.example config/config.ini
```

We recommend leaving all sections unchanged except for the IMAP and OPENVPN sections which will be covered next.

Setting up the IMAP Configuration

The AI VPN generates VPN profiles automatically. Currently users can request new VPN profiles via email by sending an email with an specific keyword: VPN.

The mod_comm_recv and mod_comm_send are the modules that uses the IMAP configuration to receive VPN requests from the users and to send new VPN profiles from the users.

We recommend using a dedicated email account to run this service. Some email providers offer APP Passwords, which give non-official apps permissions to access the email account. These passwords can be revoked at any time.

The AI VPN was tested with GMail. Google provides instructions on how to set an app password in an existing account: https://support.google.com/mail/answer/185833

Once the APP Password is generated, replace the values in the configuration file with the appropriate values.

Setting up the OPENVPN Configuration

The next step is to replace the example values of the OPEN VPN service with the IP address or host of the host machine.

Find the public IPv4 address of the host machine:

```
$ curl -4 icanhazip.com
```

Use this IP address to replace the placeholder in the configuration file:

```
$ SERVER_PUBLIC_URL = tcp://x.x.x.x
$ PKI_ADDRESS = x.x.x.x
$ NETWORK_CIDR = 192.168.254.0/24
$ DNS_SERVER = <pi-hole ip address here>
```

Setting up the WIREGUARD VPN Configuration

The next step is to replace the example values of the WireGuard VPN service with the IP address or host of the host machine.

Find the public IPv4 address of the host machine:

```
$ curl -4 icanhazip.com
```

Use this IP address to replace the placeholder in the configuration file:

```
$ SERVER_PUBLIC_URL = udp://x.x.x.x
$ PKI_ADDRESS = x.x.x.x
$ NETWORK_CIDR = 192.168.254.0/24
```

The WireGuard VPN also needs to configure certain parameters in a file called '.ENV'. First copy the file *.env_TEMPLATE* to *.env*:

```
$ cp .env_TEMPLATE .env
```

Then replace the server adress and server port with the parameters for your server (this has to match the config.ini file):

```
$ ENV_SERVERURL=<server_ip>
$ ENV_SERVERPORT=<server_port>
```

Save and exit. You are ready to run this module.

Setting up the NOVPN Configuration

The next step is to replace the example values of the OPEN VPN service without encryption with the IP address or host of the host machine.

Find the public IPv4 address of the host machine:

```
$ curl -4 icanhazip.com
```

Use this IP address to replace the placeholder in the configuration file:

```
$ SERVER_PUBLIC_URL = tcp://x.x.x.x:port
$ PKI_ADDRESS = x.x.x.x
$ NETWORK_CIDR = 192.168.254.0/24
$ DNS_SERVER = <pi-hole ip address here>
```

Setting up the AIVPN Configuration

The AIVPN follows certain restrictions regarding for how long the VPN profiles remain active, how many active VPN profiles can a user have simultanously, and others.

By default, the AIVPN will revoke issued VPN profiles every 72 hours. To extend or reduce this time, replace the value of the following parameter (in hours):

```
$ EXPIRATION_THRESHOLD = X
```

The AIVPN allows a maximum of 5 simultanous active VPN profiles per user. To increase or reduce this limit, replace the value of the following parameter:

```
$ ACTIVE_ACCOUNT_LIMIT = X
```

1.4 Creating a Telegram Bot for the AI VPN

Telegram is one of the supported technologies of the AI VPN used to request new VPN profiles. Requesting new VPN profiles via Telegram may be beneficial for a number of reasons, but mostly when individuals suspect their email is being monitored.

These steps need to be followed only if you are deploying the AI VPN on your organization. To complete this process you need to have a Telegram account already registered.

1.4.1 Registering a new bot using Telegram BotFather

The first step is to register a new Telegram bot using the Telegram BotFather bot:

1. Go to your Telegram application
2. Click on *Search for messages or users* and type *@BotFather*.
3. Click on the menu on the lower top left of the screen and select the option */newbot*
4. The *@BotFather* will ask first for the name of the bot. This is the name displayed to the users of the service. This should represent your own organization and should be such, that users will not be able to confuse your bot with a third party bot. Remember that users will connect to the VPN profiles sent here. It is your responsibility to set this data responsibly.
5. The *@BotFather* will ask then for the username of the bot. This is the username of the bot that users will search for when trying to find the service. The username should represent your organization and be such that users will not confuse this with any other service.
6. After the previous two values are shown correctly, the *@BotFather* will display a text and an access token. Please copy the token and continue on the configuration of the AI VPN

1.5 Deployment

This section explains how to build the AI VPN modules docker images and how to deploy the AI VPN service using docker-compose.

1.5.1 Build the AI VPN Container Modules

The AI VPN comes with a bash script that builds the images automatically. The script *build.sh* contains three main sections:

- Cleaning up the docker images.
- Cleaning up the AI VPN log files.
- Building the AI VPN container images using docker commands.

NOTE: The cleaning up of the docker images will remove all dangling docker images, that is, docker images that have no links or relationships with images that are tagged. Cleaning dangling images frees space. If you do not want to perform this step, comment the following line from the *build.sh* script:

```
$ docker rmi -f $(docker images -f "dangling=true" -q)
```

Run the build script to build the container images:

```
$ cd AIVPN/
$ sudo ./build.sh
```

1.5.2 Deploy the AI VPN service

The deployment of the AI VPN is done using docker-compose:

```
$ cd AIVPN/
$ sudo docker-compose -f docker-compose.yml up
```

1.5.3 Check the AI VPN Service Health

Check the AI VPN modules are running using Docker:

```
$ sudo docker ps
```

Check the AI VPN modules are working using the logs:

```
$ cd AIVPN/
$ tail -f logs/*.log
```

Check the AI VPN email configuratio works:

- Send an email to the email address used for the service with the word: *VPN* in the body or subject of the email.
- After a few minutes a new VPN profile should be received.

1.6 AI VPN Using Nix Arion

1.6.1 What is Nix?

Nix is a packet manager that uses a model that allows reproducibility of packages and builds using declarative definitions. Learn more about Nix and NixOS at *NixOS Quick Start <https://nixos.org/manual/nix/stable/quick-start.html>*.

1.6.2 What is Arion?

Arion is a Nix tool designed to help launch modular docker based applications on Nix. Arion was designed with the same Nix principles in mind and follows a declarative approach. Arion focuses on providing an easier deployment and better performance. Learn more about Arion at *Arion Documentation <https://docs.hercules-ci.com/arion/>*.

1.6.3 How can the AI VPN deployed using Nix Arion?

Once Nix package manager and Arion are already installed, you can start the service with a simple command:

```
$ cd AIVPN/
$ arion up -d
```

To stop the service run:

```
$ arion down
```

If you need to change the configuration of the services, edit the file:

```
* arion-compose.nix
```

Note: Arion has some limitations and may not support all the configuration parameters of docker-compose. This is why at the moment the Pi-Hole module is not supported through Arion.