

# Howest University of Applied Sciences

Applied Computer Science  
Server System Management

TI-S4 User Case Driven Paper  
Spring 2023



***Enabling Secure Access to Local Services: Proxy-Based  
Solutions for Overcoming NAT and Firewall Restrictions***

Ali Mola, Miguel De Backer, Mats Fiers, Liam Delagrense, Sinan Yücel



## Abstract.

NAT and firewalls are important for IP address management and network security, but they can make it difficult for clients in different networks to communicate directly. This research aims to find a way to enable communication between clients while still maintaining security and not changing router settings.

This study investigates the efficacy of proxy-server based solutions for overcoming the limitations imposed by Network Address Translation (NAT) and firewalls in network communication. By enabling clients to establish connections through a proxy server, it demonstrates the successful traversal of NAT and firewall barriers, thereby enhancing the overall connectivity and accessibility of network resources.

# 1 Table of Contents

2	Literature Study .....	1
2.1	Use of a reverse proxy .....	1
2.2	Network Address Translation (NAT) and Firewalls .....	1
2.3	Proxy-Server based solutions.....	1
2.4	HAProxy.....	1
2.5	Roxy-WI.....	1
2.6	Task Scheduler in Windows .....	1
2.7	Use of MYSQL.....	2
3	Introduction .....	2
3.1	Reasoning.....	2
3.2	Use Case .....	2
3.3	Benefits: .....	3
3.4	Implementation .....	3
4	Setup .....	4
4.1	Creating a VPS:.....	4
4.2	Accessing the VPS .....	6
4.3	Setting up the VPS.....	7
4.4	Install Roxy-WI (HAProxy GUI) .....	7
4.4.1	Installing Apache and cloning Roxy-WI repository .....	7
4.4.2	Installing dependencies .....	7
4.4.3	Configuring Apache.....	8
4.4.4	Installing additional requirements.....	9
4.5	Configuration of Roxy-WI.....	9
4.5.1	Configuring log rotation .....	9
4.5.2	Creating necessary directories.....	10
4.5.3	[OPTIONAL] Enabling TLS .....	10
4.5.4	Configuring ownership.....	11
4.5.5	Setting up MySQL for Roxy-WI.....	12
4.5.6	Creating user for Roxy-WI.....	14
4.5.7	Configuring an SSH key .....	14
4.5.8	Adding SSH key to Roxy-WI.....	16
4.5.9	Configure HAProxy as needed by Roxy-WI .....	18
4.5.10	Adjust statistics password.....	19
4.5.11	Connect Roxy-WI to the HAProxy server .....	20

5	Setting up Windows clients.....	22
5.1	How the clients will connect .....	22
5.1.1	Preparing the server .....	22
5.1.2	Preparing the Windows client.....	22
5.1.3	Giving access to the server .....	22
5.2	Create lower privilege user on Roxy-WI .....	24
5.3	Forwarding connections through the proxy .....	25
6	Extras.....	27
6.1	How to create a new SSH key? .....	27
7	Conclusion.....	27
8	Source references .....	28

## 2 Literature Study

### 2.1 Use of a reverse proxy

As the name implies, a reverse proxy does the opposite of what a forward proxy does: A forward proxy acts on behalf of clients (or requesting hosts). Forward proxies can hide the identities of clients whereas reverse proxies can hide the identities of servers. (MDN Web docs, 2023)

### 2.2 Network Address Translation (NAT) and Firewalls

Network Address Translation (NAT) is designed for IP address conservation. It enables private IP networks that use unregistered IP addresses to connect to the Internet. NAT operates on a router, usually connecting two networks together, and translates the private (not globally unique) addresses in the internal network into legal addresses, before packets are forwarded to another network. (Cisco, 2020).

### 2.3 Proxy-Server based solutions

A proxy server is an intermediary between an end user/computer and the internet. A proxy server acts just like a traffic conductor. Depending on where the proxy server lies in your network (more on this later), it will inspect and route internet traffic to/from the user and the requested web address. (Cisco Umbrella, 2023).

### 2.4 HAProxy

HAProxy, which stands for High Availability Proxy, is a popular open-source software TCP/HTTP Load Balancer and proxying solution which can be run on Linux, macOS, and FreeBSD. Its most common use is to improve the performance and reliability of a server environment by distributing the workload across multiple servers (e.g., web, application, database). It is used in many high-profile environments, including GitHub, Imgur, Instagram, and Twitter. (DigitalOcean, 2022).

### 2.5 Roxy-WI

Roxy-WI is a web interface that helps users and administrators easily set up and manage a reliable infrastructure using popular services like HAProxy, NGINX, Apache, and Keepalived. It provides a central monitoring interface to keep track of load statistics and server statuses (Roxy-WI, 2023).

### 2.6 Task Scheduler in Windows

Task Scheduler is a component of Microsoft Windows that provides the ability to schedule the launch of programs or scripts at pre-defined times or after specified time intervals. It was first introduced in the Microsoft Plus! for Windows 95 as System Agent (Microsoft, 2023).

## 2.7 Use of MYSQL

The instructions provided in this subsection follow the standard procedure for installing and configuring a MySQL database on a Linux server. The instructions explain how to create a new user, create a database, and grant privileges to the user to access the database. These instructions are well documented and can be found in various online resources (e.g., MySQL documentation, Linux documentation). The subsection also explains how to create a new user that will administer the HAProxy server. The instructions cover creating a new user account, generating an SSH key pair, and setting up the correct permissions for the new user. The instructions on editing the SSH configuration file are specific to Roxy-WI and are required for it to accept RSA keys. This information is only relevant for users of Roxy-WI and is not necessary for users of other software.

## 3 Introduction

### 3.1 Reasoning

Network Address Translation (NAT) devices are commonly used to provide internet access to clients using private IP addresses. However, this creates a challenge when external clients need to access services running on these machines. To address this issue, a reverse proxy like HAProxy can be used to forward incoming requests to the appropriate machine.

While documentation is available for HAProxy and Roxy-Wi individually, this paper offers the convenience of consolidating all the necessary information in one place. It presents a comprehensive guide on setting up and integrating HAProxy and Roxy-Wi together, providing step-by-step instructions and sharing best practices. By utilizing this paper, readers can save time and effort by accessing all the required information in a single resource for effective server system management.

### 3.2 Use Case

In a typical enterprise environment, there may be several services running on machines behind a NAT device that require access from external clients. To enable this access while maintaining security and privacy, a reverse proxy like HAProxy can be used. By routing incoming requests through the reverse proxy, external clients can access the services without directly exposing the machines to the internet.

This approach provides several benefits, including improved security, easier management, and maintenance with the downside of added complexity and slower performance. By centralizing access to the services, IT teams can more easily monitor and control traffic flow, implement security policies, and troubleshoot issues.

Additionally, this makes the needed access to firewall management unnecessary. The connection to the machine will be tunnelled through the proxy server.

### 3.3 Benefits:

#### **Improved security:**

By routing incoming requests through a reverse proxy like HAProxy, the back-end servers are shielded from direct exposure to the internet, reducing the risk of cyber-attacks and unauthorized access. The reverse proxy can also implement security policies such as access controls and authentication.

This approach also makes it unnecessary to edit firewall rules.

#### **Easier Management and Maintenance:**

Centralizing access to services behind a reverse proxy simplifies the management and maintenance of the network infrastructure. IT teams can monitor traffic flow, identify, troubleshoot issues, and update configurations more easily, reducing the time and effort required to maintain the network.

#### **TLS offloading:**

When using TLS, offloading SSL termination and other processing tasks to the reverse proxy, the back-end servers can focus on delivering content more efficiently. This results in better performance and faster response times for users accessing the services.

This however is not applicable to cases where TCP connections will be forwarded.

#### **Scalability:**

The use of a reverse proxy like HAProxy enables organizations to easily scale their services by adding more back-end servers as needed, without affecting the external clients. This makes it easier to accommodate increasing traffic and demand for the services.

#### **Cost-Effective:**

Using a reverse proxy like HAProxy is a cost-effective solution compared to other alternatives like VPNs or leased lines. It eliminates the need for additional hardware or software, reducing capital and operating expenses.

### 3.4 Implementation

To implement a reverse proxy using HAProxy, a Linux machine with root access is needed. In this use case, a Virtual Private Server (VPS) running Ubuntu 22.10 will be used. A VPS is a type of service that uses virtualization technology to create a virtual machine on a physical server. This virtual machine is isolated from other virtual machines on the same physical server, giving it the appearance and functionality of a dedicated server. The following steps outline the basic process for installing and configuring HAProxy on a Linux machine.



## 4 Setup

### 4.1 Creating a VPS:

A VPS provides customers with dedicated resources and full control over the operating system and applications installed on the server. DigitalOcean is a popular cloud hosting provider that offers an easy-to-use interface for creating and managing VPS instances. Once a DigitalOcean account is created, spin up a VPS instance in minutes and start configuring it according to the needs.

1. Sign up for DigitalOcean if there is no account available.
2. Once created, the homepage will show a default project. On this page users can create Droplets, which are the name DigitalOcean uses for VPS's.
3. At the top right, there is a button 'Create'.

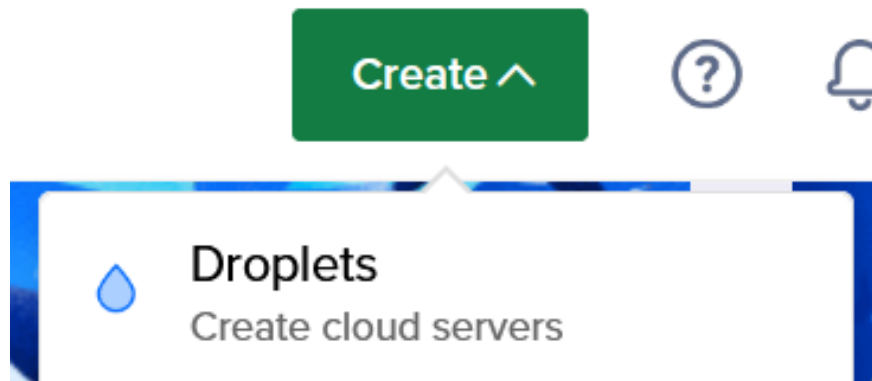


Figure 1: Creating a Droplet.

4. Choose 'Droplet' from the list.
5. Depending on the location, choose the best/closest server.
6. Make sure the following options are configured to the needs:
  - a. The desired datacentre.
  - b. The desired OS image and resources.

CPU options

Regular Disk type: SSD	Premium Intel Disk: NVMe SSD	Premium AMD Disk: NVMe SSD
<b>\$6/mo</b> \$0.009/hour	<b>\$12/mo</b> \$0.018/hour	<b>\$18/mo</b> \$0.027/hour
1 GB / 1 CPU 25 GB SSD Disk 1000 GB transfer	2 GB / 1 CPU 50 GB SSD Disk 2 TB transfer	2 GB / 2 CPUs 60 GB SSD Disk 3 TB transfer
		<b>\$24/mo</b> \$0.036/hour
		4 GB / 2 CPUs 80 GB SSD Disk 4 TB transfer
		<b>\$48/mo</b> \$0.071/hour
		8 GB / 4 CPUs 160 GB SSD Disk 5 TB transfer
		<b>\$96/mo</b> \$0.143/hour
		16 GB / 8 CPUs 320 GB SSD Disk 6 TB transfer

Figure 2: Choosing a plan.

- c. Choose an authentication method. SSH Keys are preferred. More information about creating SSH keys can be found at [5.1](#) There is also an option to use a password. This however is not recommended as the server will be publicly accessible and is less secure.

### Choose Authentication Method ?

☒ **SSH Key**  
Connect to your Droplet with an SSH key pair

☐ **Password**  
Connect to your Droplet as the "root" user via password

Choose your SSH keys

☐ **Select all** ☒ **HAProxy**

New SSH Key

Figure 3: Choosing the authentication method.

d. Choose a hostname.

### Finalize Details

**Quantity**  
Deploy multiple Droplets with the same configuration.

—

1 Droplet

+

**Hostname**  
Give your Droplets an identifying name you will remember them by.

haproxy

**Tags**


Type tags here

**Project**

HAProxy

Figure 4: Setting a hostname for the server



- e. Click on create Droplet.
- f. After the installation process is done, there will be an IP address to connect to.

 **HAProxy** DEFAULT  
Other / testing env


→ Move Resources


Resources Activity Settings

DROPLETS (1)

  haproxy

167.99.45.94

+ 

+ 

...

Figure 5: Droplet with its IP address

## 4.2 Accessing the VPS

To access the VPS, it will be necessary to establish an SSH connection using the command line. Since the public SSH key has been added to the VPS during creation, it can be used to securely authenticate ourselves to the VPS. If the password option was chosen, use the chosen password.

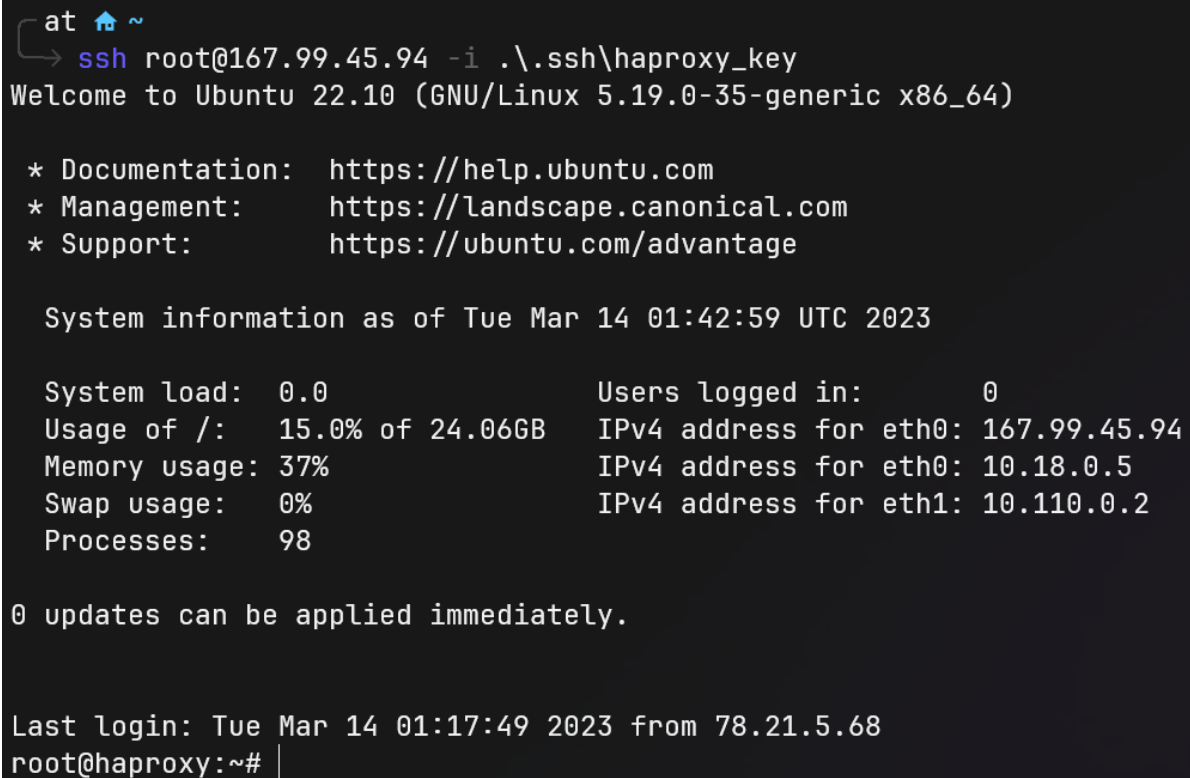
To access the VPS, follow these steps:

1. Open a shell on the local machine.
2. Connect with this command:

```
ssh root@ip-address-or-hostname-of-vps
```

Replace ip-address-or-hostname-of-vps with the public IP address or domain name assigned the VPS instance.

3. Press Enter and wait for the SSH connection to be established with the VPS.



```
at 🏠 ~  
→ ssh root@167.99.45.94 -i ~/.ssh\haproxy_key  
Welcome to Ubuntu 22.10 (GNU/Linux 5.19.0-35-generic x86_64)  
  
* Documentation:  https://help.ubuntu.com  
* Management:    https://landscape.canonical.com  
* Support:       https://ubuntu.com/advantage  
  
System information as of Tue Mar 14 01:42:59 UTC 2023  
  
System load:  0.0           Users logged in:      0  
Usage of /:   15.0% of 24.06GB IPv4 address for eth0: 167.99.45.94  
Memory usage: 37%          IPv4 address for eth0: 10.18.0.5  
Swap usage:   0%           IPv4 address for eth1: 10.110.0.2  
Processes:   98  
  
0 updates can be applied immediately.  
  
Last login: Tue Mar 14 01:17:49 2023 from 78.21.5.68  
root@haproxy:~# |
```

Figure 6: Successful SSH login

### 4.3 Setting up the VPS

When a fresh server is created, it is always highly recommended to update the system. To do so, use the following command for Debian based systems:

```
apt update && apt upgrade -y
```

Install haproxy

```
apt install haproxy -y
```

### 4.4 Install Roxy-WI (HAProxy GUI)

The installation of Roxy-WI is not too difficult but has its caveats. This section tries to explain it as thoroughly as possible. For more information the official documentation of Roxy-WI can be used:

[Installation Roxy-WI](#)

#### 4.4.1 Installing Apache and cloning Roxy-WI repository

First, apache2 is needed.

```
apt install apache2 -y
```

Clone the git repository of Roxy-WI to /var/www/

```
git clone https://github.com/hap-wi/roxy-wi.git /var/www/haproxy-wi
```

#### 4.4.2 Installing dependencies

All the needed dependencies are listed by the creators of Roxy-WI. Install them by running the following command:

```
apt install -y python3 python3-pip python3-ldap rsync ansible  
python3-requests python3-networkx python3-matplotlib python3-bottle  
python3-future python3-jinja2 python3-peewee python3-distro python3-  
pymysql python3-psutil python3-paramiko netcat-traditional nmap net-  
tools lshw dos2unix libapache2-mod-wsgi-py3 openssl sshpass
```

### 4.4.3 Configuring Apache

After installing all the dependencies, make sure that the current folder is `/var/www/` folder. If not, type the following command:

```
cd /var/www/
```

The cloned folder 'haproxy-wi' must be owned by www-data. Setting the ownership of the web application files and directories to www-data ensures that the web server software can access and execute the code, while keeping it secure from other users on the system.

```
chown www-data:www-data -R haproxy-wi/
```

Copy the HAProxy config to Apache (sites-available). The roxy-wi.conf file contains Apache configuration directives that define how requests to the Roxy-WI interface should be handled by the web server. By copying the file to the sites-available directory and renaming it, it is made available to Apache for activation.

```
cp haproxy-wi/config_other/httpd/roxy-wi_deb.conf  
/etc/apache2/sites-available/roxy-wi.conf
```

Enable the copied configuration file.

```
a2ensite roxy-wi.conf
```

Now enable some modules to make Roxy-WI function correctly. The following modules need to be enabled:

modulename	function
cgid	This module provides support for executing CGI scripts using the CGI daemon, which can improve performance and security.
ssl	This module provides support for SSL/TLS encryption, which is essential for securing web traffic over HTTPS.
proxy_http	This module provides support for proxying HTTP requests to other servers, which can be useful for load balancing or caching.
rewrite	This module provides support for URL rewriting, which allows modifying URLs or redirect traffic based on certain rules.

```
a2enmod cgid ssl proxy_http rewrite
```

#### 4.4.4 Installing additional requirements

Well, of course there are some requirements to install. Again Roxy-WI creators have collected them for us. It can be found in the txt file under haproxy-wi/config\_other/ as requirements\_deb.txt

**[WARNING]** The pip3 commands need to be run with elevated privileges.

```
pip3 install -r haproxy-wi/config_other/requirements_deb.txt
```

*paramiko-ng is a fork of the paramiko SSH module, which provides an implementation of the SSH protocol in Python. The paramiko-ng package is used to create SSH connections, execute commands on remote machines, and transfer files over SSH.*

```
pip3 install paramiko-ng
```

After these changes, a restart of the webserver is needed.

```
systemctl restart apache2
```

Now set all python files in the /haproxy-wi/app folder as executable by adding the execute permission:

```
chmod +x haproxy-wi/app/*.py
```

## 4.5 Configuration of Roxy-WI

### 4.5.1 Configuring log rotation

In Unix-like operating systems, log rotation is a process of managing log files by compressing, renaming, and deleting old log files to prevent them from taking up too much disk space. The logrotate utility is commonly used for this purpose and is configured using configuration files in the /etc/logrotate.d/ directory.

```
cp haproxy-wi/config_other/logrotate/* /etc/logrotate.d/
```

#### 4.5.2 Creating necessary directories

Now, create some necessary directories/folders.

command	function
<code>mkdir /var/lib/roxy-wi/</code>	creates a directory called roxy-wi inside the /var/lib directory, which is typically used for storing variable data files.
<code>mkdir /var/lib/roxy-wi/keys/</code>	creates a sub-directory called keys inside the roxy-wi directory. This directory can be used for storing SSL keys and certificates.
<code>mkdir /var/lib/roxy-wi/configs/</code>	creates a sub-directory called configs inside the roxy-wi directory. This directory can be used for storing configuration files for various applications.
<code>mkdir /var/lib/roxy-wi/configs/haproxyconfig/</code>	creates a sub-directory called haproxyconfig inside the configs directory. This directory can be used for storing HAProxy configuration files.
<code>mkdir /var/lib/roxy-wi/configs/kpconfig/</code>	creates a sub-directory called kpconfig inside the configs directory. This directory can be used for storing Keepalived configuration files.
<code>mkdir /var/lib/roxy-wi/configs/nginxconfig/</code>	creates a sub-directory called nginxconfig inside the configs directory. This directory can be used for storing Nginx configuration files.
<code>mkdir /var/lib/roxy-wi/configs/apacheconfig/</code>	creates a sub-directory called apacheconfig inside the configs directory. This directory can be used for storing Apache configuration files.
<code>mkdir /var/log/roxy-wi/</code>	creates a directory called roxy-wi inside the /var/log directory, which is typically used for storing log files.
<code>mkdir /etc/roxy-wi/</code>	creates a directory called roxy-wi inside the /etc directory, which is typically used for storing system configuration files. This directory can be used for storing configuration files for the Roxy-WI web interface.

```
mkdir /var/lib/roxy-wi/
mkdir /var/lib/roxy-wi/keys/
mkdir /var/lib/roxy-wi/configs/
mkdir /var/lib/roxy-wi/configs/hap_config/
mkdir /var/lib/roxy-wi/configs/kp_config/
mkdir /var/lib/roxy-wi/configs/nginx_config/
mkdir /var/lib/roxy-wi/configs/apache_config/
mkdir /var/log/roxy-wi/
mkdir /etc/roxy-wi/
```

Move the roxy-wi.cfg file to the /etc/roxy-wi directory. This configuration file is used by the Roxy-WI web interface to define its settings and behaviour.

```
mv haproxy-wi/roxy-wi.cfg /etc/roxy-wi
```

#### 4.5.3 [OPTIONAL] Enabling TLS

After all those steps, a TLS certificate can be created to secure the connection to the website. To do so, this guide can be followed:

[DigitalOcean's Guide on how to use Let's Encrypt SSL certificate](#)

#### 4.5.4 Configuring ownership

What comes next is taking ownership of some folders. www-data must be the owner of those folders to function properly.

Folders that need changes:

folder name	function
/var/www/haproxy-wi/	This directory is the root directory for the HAProxy-WI web interface.
/var/lib/roxy-wi/	This directory is used by the Roxy-WI web interface to store various data files, such as configuration files.
/var/log/roxy-wi/	This directory is used by the Roxy-WI web interface to store log files.
/etc/roxy-wi/	This directory is used by the Roxy-WI web interface to store configuration files.

```
chown -R www-data:www-data /var/www/haproxy-wi/
chown -R www-data:www-data /var/lib/roxy-wi/
chown -R www-data:www-data /var/log/roxy-wi/
chown -R www-data:www-data /etc/roxy-wi/
```

After changing ownership of the directories, make systemd aware of the changes and new configurations.

```
systemctl daemon-reload
systemctl restart apache2
systemctl restart rsyslog
```



#### 4.5.5 Setting up MySQL for Roxy-WI

##### 4.5.5.1 Enabling MySQL in Roxy-WI config

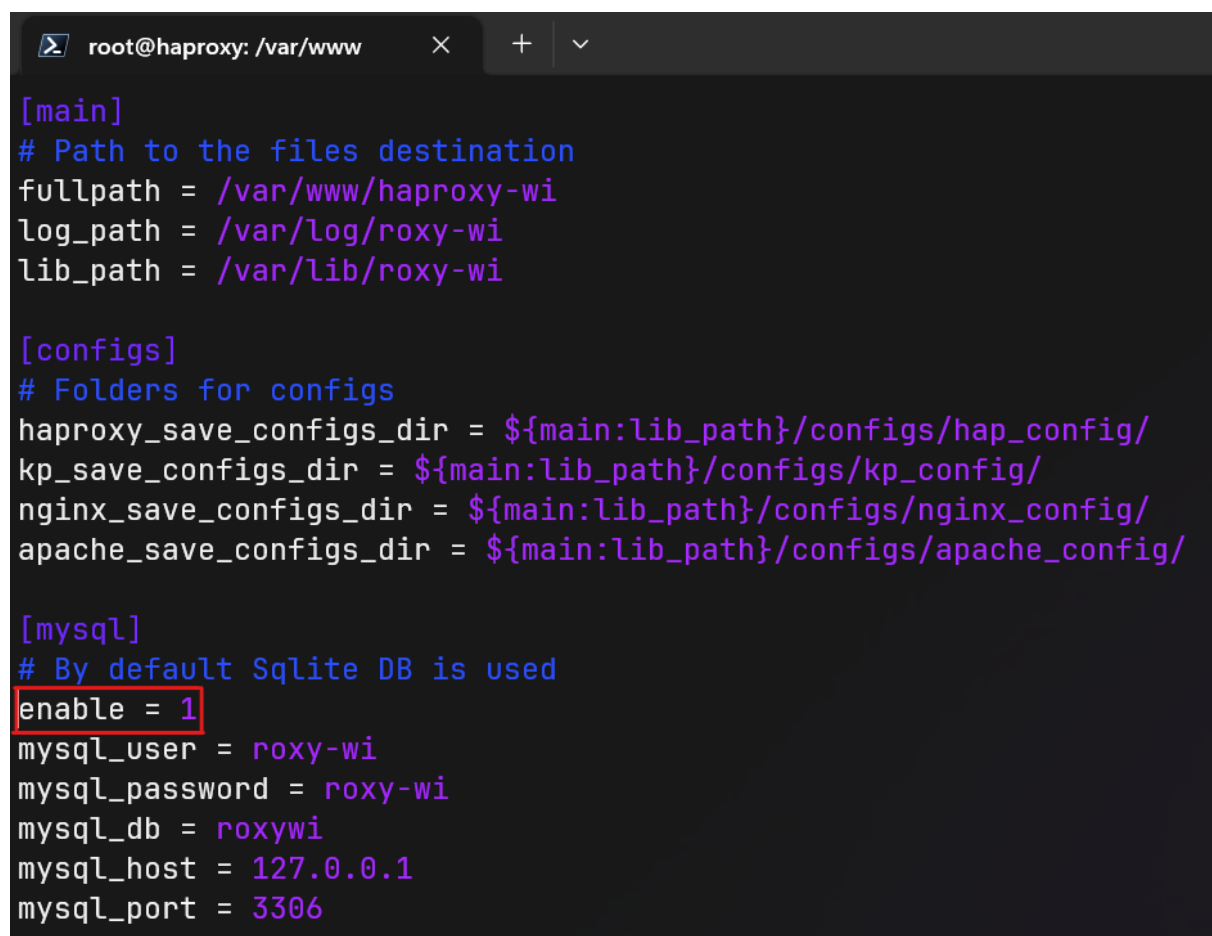
By default, Roxy-WI uses SQLite. But MySQL is recommended as it's been observed that it's more stable when installing Roxy-WI. Change the value from 0 to 1 in the config file.

There are various text editors to do this in Linux, any of them can be used. For beginners, nano is recommended. In this paper vim will be used.

```
vim /etc/roxy-wi/roxy-wi.cfg
```

change the line *enable* in the [mysql] section to '1'.

**[OPTIONAL]** change the database user password.



```
root@haproxy: /var/www
[main]
# Path to the files destination
fullpath = /var/www/haproxy-wi
log_path = /var/log/roxy-wi
lib_path = /var/lib/roxy-wi

[configs]
# Folders for configs
haproxy_save_configs_dir = ${main:lib_path}/configs/hap_config/
kp_save_configs_dir = ${main:lib_path}/configs/kp_config/
nginx_save_configs_dir = ${main:lib_path}/configs/nginx_config/
apache_save_configs_dir = ${main:lib_path}/configs/apache_config/

[mysql]
# By default Sqlite DB is used
enable = 1
mysql_user = roxy-wi
mysql_password = roxy-wi
mysql_db = roxywi
mysql_host = 127.0.0.1
mysql_port = 3306
```

Figure 7: Enabling MySQL

##### 4.5.5.2 Installing and configuring MySQL

Install MySQL

```
apt install mysql-server -y
```

The next step is creating the MySQL user. First, open the MySQL cli:

*(This command must be run as root)*

```
mysql
```

If the MySQL password was changed in the config file, please enter that password within the command:

```
CREATE USER 'roxy-wi'@'localhost' IDENTIFIED WITH  
mysql_native_password BY 'roxy-wi';
```

Now create the database.

```
CREATE DATABASE roxywi; GRANT ALL PRIVILEGES ON roxywi.* TO 'roxy-  
wi'@'localhost';
```

To exit MySQL cli:

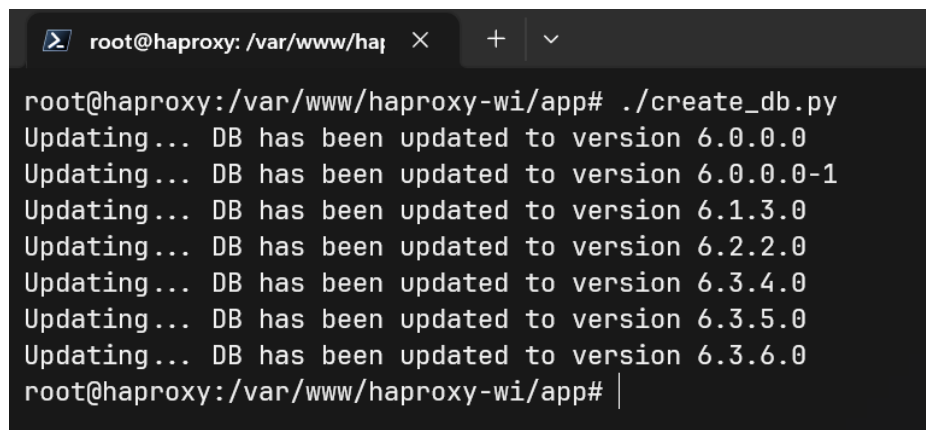
```
exit
```

#### 4.5.5.3 Create default databases

There is a MySQL script in Python written by the Roxy-WI creators. It's used to populate the database which is necessary for Roxy-WI to function properly.

```
cd /var/www/haproxy-wi/app  
./create_db.py
```

The output should look like this:

A terminal window with a dark background. The title bar shows 'root@haproxy: /var/www/hap' with window control buttons. The terminal text shows the command './create\_db.py' being executed, followed by seven lines of output: 'Updating... DB has been updated to version 6.0.0.0', 'Updating... DB has been updated to version 6.0.0.0-1', 'Updating... DB has been updated to version 6.1.3.0', 'Updating... DB has been updated to version 6.2.2.0', 'Updating... DB has been updated to version 6.3.4.0', 'Updating... DB has been updated to version 6.3.5.0', and 'Updating... DB has been updated to version 6.3.6.0'. The prompt 'root@haproxy:/var/www/haproxy-wi/app#' is shown at the end with a cursor.

```
root@haproxy:/var/www/haprox... x + v  
root@haproxy:/var/www/haproxy-wi/app# ./create_db.py  
Updating... DB has been updated to version 6.0.0.0  
Updating... DB has been updated to version 6.0.0.0-1  
Updating... DB has been updated to version 6.1.3.0  
Updating... DB has been updated to version 6.2.2.0  
Updating... DB has been updated to version 6.3.4.0  
Updating... DB has been updated to version 6.3.5.0  
Updating... DB has been updated to version 6.3.6.0  
root@haproxy:/var/www/haproxy-wi/app# |
```

Figure 8: Creating databases

#### 4.5.5.4 Check the deployment

Test if everything works by surfing to the hostname of the VPS and logging in using the default credentials: `\admin:admin'`

If successful, there should be a dashboard.

**Note:** The default `roxy-wi.conf` only makes `apache2` respond with the dashboard on port 443

#### 4.5.6 Creating user for Roxy-WI

Now create a user that will administer the HAProxy server.

***The username 'proxyadmin' will be used in this paper. But it's not necessary, any username may be used.***

Create the user 'proxyadmin':

```
useradd -m proxyadmin
```

#### 4.5.7 Configuring an SSH key

SSH keys are used to allow Roxy-WI to connect the machine that has HAProxy installed.

Use this command to generate a new SSH keypair.

```
sudo -u proxyadmin ssh-keygen -t rsa -f /home/proxyadmin/.ssh/id_rsa  
-q -N ""
```

The command above contains many parameters. Here is a breakdown:

parameter	meaning
-u proxyadmin	This option specifies that the subsequent command should be run as the user "proxyadmin".
ssh-keygen	This command is used to generate an SSH key pair for a user.
-t rsa	This option specifies the type of key pair to generate. In this case, it is RSA. Roxy-WI only accepts RSA keys.
-f /home/proxyadmin/.ssh/id_rsa	This option specifies the filename of the private key to be generated. The private key will be stored in the "/home/proxyadmin/.ssh/" directory with the filename "id_rsa"
-q	This option specifies that ssh-keygen should run in quiet mode, which means it will not prompt the user for input or output messages.
-N ""	This option specifies that the private key should not be password-protected. The empty string means that no passphrase will be required to use the key.

After that add the public key to the authorized\_keys file. The >> sign means concatenate into that file. Also, if there is no file named authorized\_keys, it will be automatically created.

```
sudo -u proxyadmin cat /home/proxyadmin/.ssh/id_rsa.pub >>
/home/proxyadmin/.ssh/authorized_keys
```

Then, the owner of the authorized\_keys file must be the new user itself and the correct permissions must be given to that file.

```
chown proxyadmin:proxyadmin /home/proxyadmin/.ssh/authorized_keys
sudo -u proxyadmin chmod 600 /home/proxyadmin/.ssh/authorized_keys
```

Before adding the key in Roxy-WI, give the users the correct permissions to manage the HAProxy server. Otherwise, it will not have enough permissions to manage the HAProxy server.

Type the following command to edit the /etc/sudoers file:

```
visudo
```

Add these lines to the bottom of the file:

```
www-data ALL=(ALL) NOPASSWD:ALL
proxyadmin ALL=(ALL) NOPASSWD:ALL
```

**[NOTE]** This is not best-practice. Giving users the minimum privileges they need to complete their tasks would be optimal. Unfortunately, Roxy-WI has no documentation about which commands it uses in elevated privileges. So, this is a simple solution to this problem, but be aware of the risks!

Then restart Apache2 to apply the changes.

```
systemctl restart apache2
```

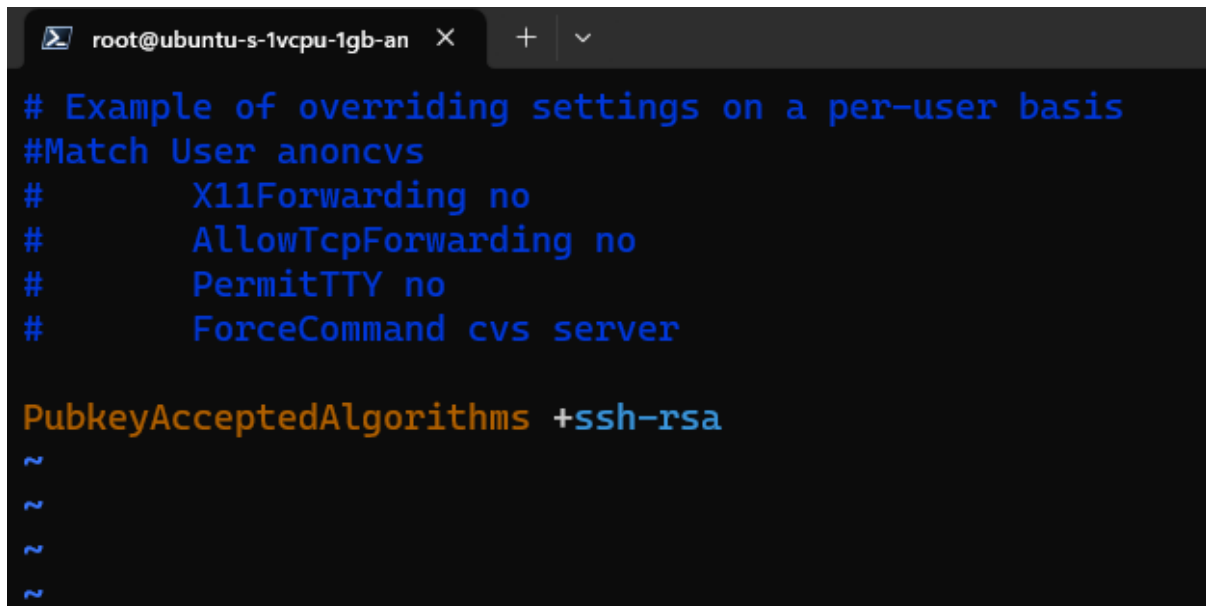
Edit the ssh config:

```
vim /etc/ssh/sshd_config
```

Add this to the end of the file:

```
PubkeyAcceptedAlgorithms +ssh-rsa
```

**[INFO]** This is needed because Roxy-WI or a library it uses only supports RSA keys.



```
root@ubuntu-s-1vcpu-1gb-an X + v
# Example of overriding settings on a per-user basis
#Match User anoncvs
#       X11Forwarding no
#       AllowTcpForwarding no
#       PermitTTY no
#       ForceCommand cvs server

PubkeyAcceptedAlgorithms +ssh-rsa
~
~
~
~
```

Figure 9: Configuring SSH

After this restart the ssh service to apply the changes:

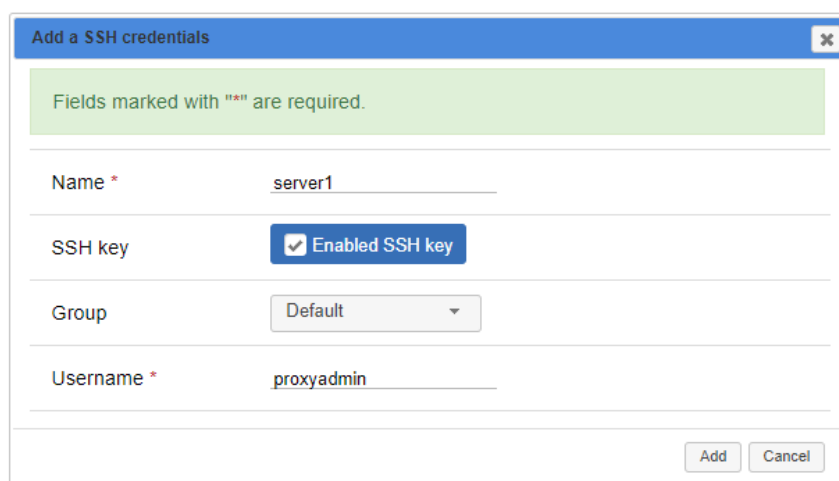
```
systemctl restart ssh
```

#### 4.5.8 Adding SSH key to Roxy-WI

Navigate to the Roxy-WI gui, login using 'admin:admin'

Navigate to SSH credentials:

Servers → SSH credentials



The dialog box titled "Add a SSH credentials" contains a green message box stating "Fields marked with '\*' are required." Below this, there are four input fields: "Name \*" with the value "server1", "SSH key" with a checked checkbox and the label "Enabled SSH key", "Group" with a dropdown menu showing "Default", and "Username \*" with the value "proxyadmin". At the bottom right, there are "Add" and "Cancel" buttons.

Figure 10: Configure added SSH key

Let's read and copy our private key in Roxy-WI.

**[WARNING] DO NOT SHARE THIS KEY WITH ANYONE!**

```
cat /home/proxyadmin/.ssh/id_rsa
```

Copy the output and add the key:

The screenshot shows the 'SSH Credentials' page in the Roxy-WI interface. At the top, there's a table with columns 'Name', 'SSH key', and 'Group'. The first row shows 'server1\_Default' with an 'Enable SSH key' button and a 'Default' group dropdown. Below this, the 'Upload SSH key' section is active. It features a dropdown menu for the key name (currently 'server1\_Default') and a text area for the key content. The text area contains a sample private key: '-----BEGIN OPENSSH PRIVATE KEY-----\nb3BlbnNzaC1rZXktbjEAAAAAAAAABG5vbmUAAAAEbm9uZQAAAAAAAAABAAABlwAAAAadzcz2gtcn\nNhAAAAAwEAAQAAAYEAgoj9UGnkE6UkwD8Z3+KsQ\nYFMht7OhRE7OJ9tuRLHcNjdjO7GT3t'. An 'Upload' button is located below the text area. At the bottom, a light blue informational banner states: 'You can read the description of all parameters [here](#), read HowTo in this [article](#)'.

Figure 11: Add private key to Roxy-WI

If this screen appears, the key has been successfully created:

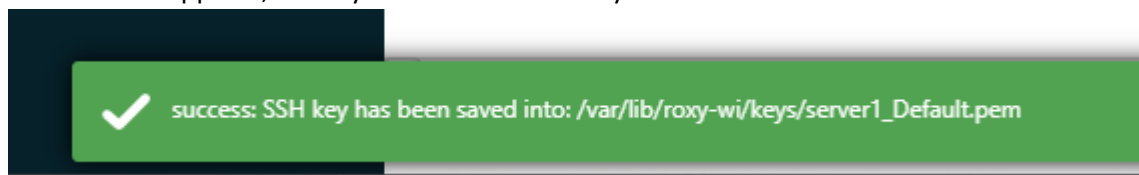


Figure 12: Success message

#### 4.5.9 Configure HAProxy as needed by Roxy-WI

Edit the configuration file:

```
vim /etc/haproxy/haproxy.cfg
```

Add or change these values in the file:

```
global

    stats socket /var/lib/haproxy/stats
    stats socket *:1999 level admin
    stats socket /var/run/haproxy.sock mode 600 level admin
    server-state-file /etc/haproxy/haproxy.state

defaults

    load-server-state-from-file global
    mode tcp
    option tcplog

listen stats

    bind *:8085
    mode http
    stats enable
    stats uri /stats
    stats realm HAProxy-04\ Statistics
    stats auth admin:<CHANGE ME>
    stats admin if TRUE
```

Change the statistics page password as desired. This will later also be changed in the Roxy-WI GUI

The file should look like this:

```
:DHE-RSA-AES128-GCM-SHA256:DHE-RSA-AES256-GCM-SHA384
ssl-default-bind-ciphersuites TLS_AES_128_GCM_SHA256:TLS
05_SHA256
ssl-default-bind-options ssl-min-ver TLSv1.2 no-tls-tick
stats socket /var/lib/haproxy/stats
stats socket *:1999 level admin
stats socket /var/run/haproxy.sock mode 600 level admin
server-state-file /etc/haproxy/haproxy.state

defaults
    load-server-state-from-file global
    log global
    mode tcp
    option tcplog
    option dontlognull
    timeout connect 5000
    timeout client 50000
    timeout server 50000
    errorfile 400 /etc/haproxy/errors/400.http
    errorfile 403 /etc/haproxy/errors/403.http
    errorfile 408 /etc/haproxy/errors/408.http
    errorfile 500 /etc/haproxy/errors/500.http
    errorfile 502 /etc/haproxy/errors/502.http
    errorfile 503 /etc/haproxy/errors/503.http
    errorfile 504 /etc/haproxy/errors/504.http

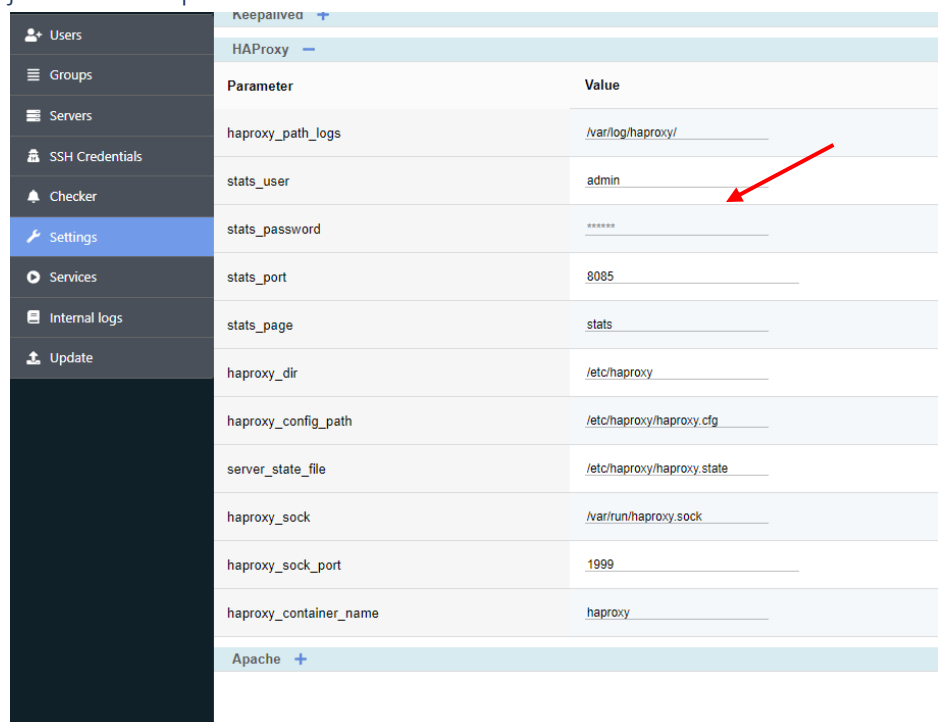
listen stats
    bind *:8085
    mode http
    stats enable
    stats uri /stats
    stats realm HAProxy-04\ Statistics
    stats auth admin:password
    stats admin if TRUE
```

Figure 13: HAProxy config file

Restart the HAProxy service:

```
systemctl restart haproxy.service
```

#### 4.5.10 Adjust statistics password



The screenshot shows the HAProxy Admin area menu. On the left is a sidebar with navigation options: Users, Groups, Servers, SSH Credentials, Checker, Settings (highlighted), Services, Internal logs, and Update. The main content area is titled 'HAProxy' and contains a table of configuration parameters. A red arrow points to the 'stats\_password' field, which is currently set to '\*\*\*\*\*'.

Parameter	Value
haproxy_path_logs	/var/log/haproxy/
stats_user	admin
stats_password	*****
stats_port	8085
stats_page	stats
haproxy_dir	/etc/haproxy
haproxy_config_path	/etc/haproxy/haproxy.cfg
server_state_file	/etc/haproxy/haproxy.state
haproxy_sock	/var/run/haproxy.sock
haproxy_sock_port	1999
haproxy_container_name	haproxy

Figure 14: Admin area menu



#### 4.5.11 Connect Roxy-WI to the HAProxy server

Now in the GUI navigate to the servers in the admin tab and add a new server:

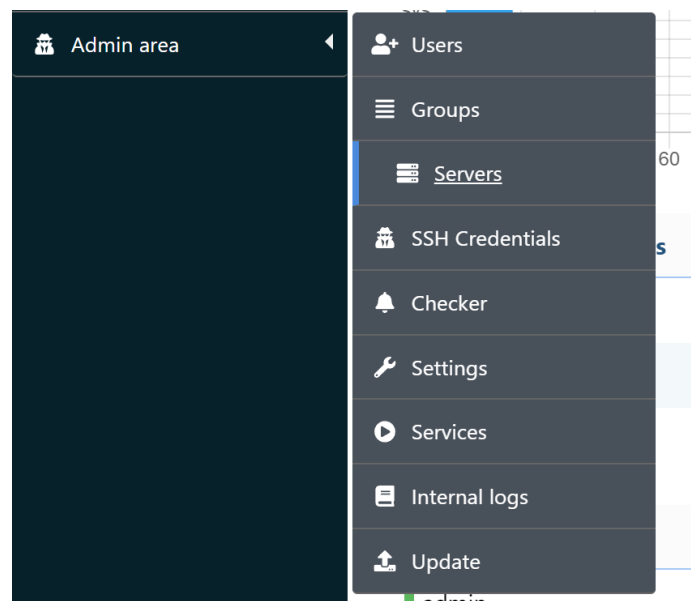


Figure 15: Admin area menu

Fill in the details of the HAProxy server:

Add a new server	
Fields marked with * are required.	
Name *	Server1
IP *	127.0.0.1
Port *	22
Enable	<input checked="" type="checkbox"/>
Virt	<input type="checkbox"/>
Scan the server	<input checked="" type="checkbox"/>
Slave for	
Description	
Credentials *	server1_Default
Group *	Default
<input type="button" value="Add"/> <input type="button" value="Cancel"/>	

Figure 16: New server settings

**[INFO]** The HAProxy server that is being managed is also installed on the same machine as Roxy-WI, so just enter the localhost address 127.0.0.1

Users Groups Servers SSH credentials Checker OpenVPN Settings Services Update Backup Monitoring installation													
/ Servers -> Servers													
Name	IP	Port	Group	Enabled	Virt	HAProxy	Nginx	Apache	Firewalld	Protected	Slave for	Cred	
server1	127.0.0.1	22	Default	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	view	<input type="checkbox"/>	Not slave	sen

Figure 17: Server added to list

Now that things are getting serious, let's change the default admin password.

Click on the admin icon in the top right corner.

Test by logging out and back in with the new password.

That concludes the setup of the proxy server. The next chapter will discuss the configuration of the windows clients, including their connection to the server and services.

## 5 Setting up Windows clients

This section discusses the setup process of the Windows clients, which enables the exposure of their local services to the wider internet using the HAProxy server.

### 5.1 How the clients will connect

#### 5.1.1 Preparing the server

The administrators of the HAProxy server will need to add a new Linux user to the server. This is done for extra security so that not everyone has root access.

```
useradd -m <proxy user>
```

This will create a new user with a home directory. This is important because it will hold the SSH keys.

#### 5.1.2 Preparing the Windows client

The clients will need to generate a new public-private keypair on their Windows machines.

```
ssh-keygen -t ed25519
```

When the command is executed, users may encounter multiple prompts. To proceed with the default settings, simply press the 'Enter' key.

#### 5.1.3 Giving access to the server

Now on the server, the `.ssh` folder needs to be created with the correct permissions that will hold the `authorized_keys` file inside it.

This command will be executed as the proxy user which was created in [5.1.1](#)

```
sudo -u <proxy user> sh -c 'mkdir -p ~/.ssh && chmod 700 ~/.ssh && touch ~/.ssh/authorized_keys && chmod 600 ~/.ssh/authorized_keys'
```

Despite the current year being 2023, it is unfortunate that Windows still lacks a native `ssh-copy-id` command, which requires the administrator to manually upload the key. However, in this particular use case, manual uploading is necessary anyway due to the absence of a user password, rendering the lack of a native command inconsequential.

The following command will read the public key created on the **Windows** client. This key needs to be copied to the users' `authorized_keys` file on the server.

```
cat "$env:USERPROFILE\.ssh\id_ed25519.pub"
```

The following command will add the copied public key from the Windows client to the users' `authorized_keys` file. This will allow the user to connect to the server using their private key on their machine.

```
echo "<generated public key>" >> /home/<proxy user>/.ssh/authorized_keys
```

Now test the connection by executing the following command on the Windows client:

```
ssh <proxy user>@<hostname of proxy server>
```

The following PowerShell script makes it easier to connect the Windows client to the desired server. Copy this script and paste it in a file with the `.ps1` extension and execute it.

```
# Check for admin perms
if (-not ([Security.Principal.WindowsPrincipal]
[Security.Principal.WindowsIdentity]::GetCurrent()).IsInRole([Security.Principi
al.WindowsBuiltInRole]::Administrator)) {
    Write-Host "This script needs to be run as an administrator."
    exit
}

$TaskName = Read-Host "Enter task name"
$Description = Read-Host "Enter task description"
$Command = "ssh.exe"
$PortLocal = Read-Host "Enter the local port you want to be forwarded"
$PortServer = Read-Host "Enter the local port you want on the server to be
listened"
$HostnameServer = Read-Host "Enter the hostname of the server"
$UsernameServer = Read-Host "Enter the username of the user on the server"
$Arguments = "-R $PortServer" + ":127.0.0.1:" + "$PortLocal
$UsernameServer@$HostnameServer"
$UserName = (Get-CimInstance -ClassName Win32_ComputerSystem).UserName
$Action = New-ScheduledTaskAction -Execute $Command -Argument $Arguments
$Trigger = New-ScheduledTaskTrigger -AtLogon -User $UserName
$Settings = New-ScheduledTaskSettingsSet -AllowStartIfOnBatteries -
DontStopIfGoingOnBatteries -DontStopOnIdleEnd -StartWhenAvailable -
RestartInterval (New-TimeSpan -Minutes 1) -RestartCount 3 -ExecutionTimeLimit
(New-TimeSpan -Seconds 0)
$Principal = New-ScheduledTaskPrincipal -UserId $UserName -LogonType S4U

Register-ScheduledTask -Action $Action -Trigger $Trigger -TaskName $TaskName -
Description $Description -Settings $Settings -Principal $Principal

# Run the task immediately
Start-ScheduledTask -TaskName $TaskName
Write-Host "Task '$TaskName' created and started."
```

This script creates a scheduled task in Windows to run a SSH command that forwards a local port to a remote server at logon. The script prompts the user for various inputs such as the task name, task description, local port number, server hostname, server username, and server port number. The script then uses this information to create a scheduled task that runs the SSH command with the specified arguments when the user logs on to the computer. The task is created with various settings and permissions to ensure it runs smoothly and does not interfere with other tasks. Finally, the script starts the scheduled task immediately and notifies the user that the task has been created and started.

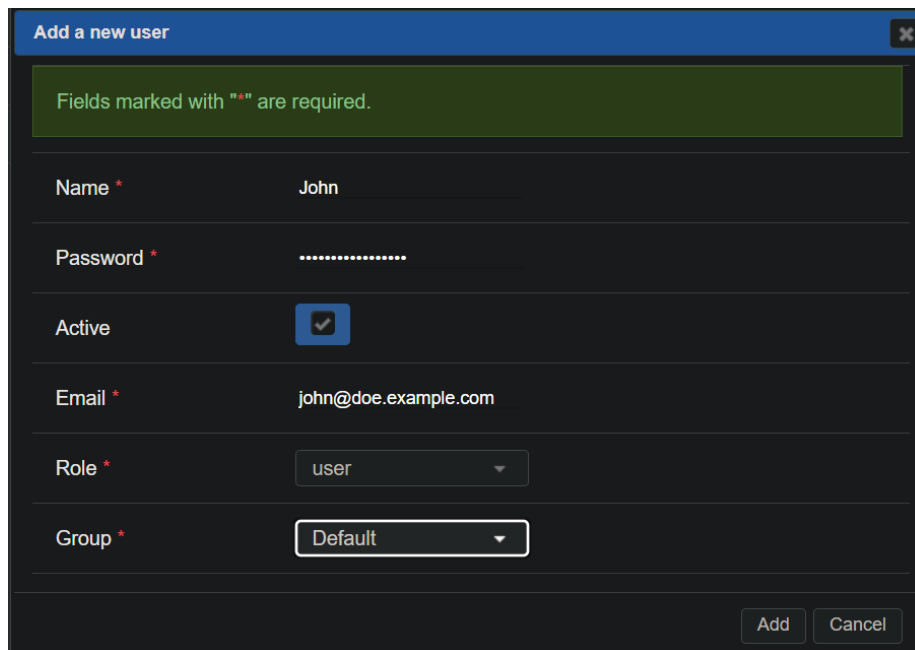
## 5.2 Create lower privilege user on Roxy-WI

Creating a new lower privilege user on the Roxy-WI GUI is important for security purposes. By default, the Roxy-WI GUI uses the "admin" account to access and manage all servers. Creating a new user with lower privileges can help to reduce the risk of unauthorized access or accidental changes to the proxy server's configuration.

The first step is to login using the admin account on the Roxy-WI GUI.

After this, navigate to the *admin area* and select *Users*.

Here a new user can be added as described on figure 18.



The screenshot shows a web form titled "Add a new user". At the top, a green banner states "Fields marked with "\*" are required." The form fields are as follows:

Field	Value
Name *	John
Password *	.....
Active	<input checked="" type="checkbox"/>
Email *	john@doe.example.com
Role *	user
Group *	Default

At the bottom right of the form are two buttons: "Add" and "Cancel".

Figure 18: Adding a new user

### 5.3 Forwarding connections through the proxy

For the configuration of the connections, login on the Roxy-WI GUI using the newly created user. Navigate to *HAProxy* and *Add proxy*.

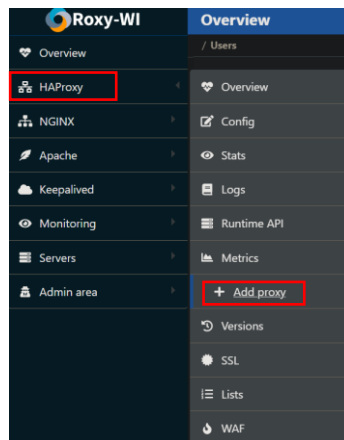


Figure 19: Add proxy

After this click on *Create Listener* and fill in the correct configuration:

**Add proxy: Create proxy**

/ Overview -> Overview -> Add proxy

Create proxy | Listener | Frontend | Backend | SSL certificates | Params | Servers | Userlists | Peers | Lists

**Add listener**

Select a server: Server1

Note: If you reconfigure Master server, Slave will be reconfigured automatically

Name: web\_8080

IP and Port: Any : 8080 +

If the IP-address for the listener is empty, it will listen on all IP addresses. Start typing IP or press down button. Click on + to add using VRRP, leave the IP field blank. If you assign a VRRP IP, the slave server will not start.

Mode: tcp

Maxconn: 2000

This value should not exceed the global maxconn. Default global maxconn value: 2000

Balance: roundrobin

Health check: Choose a custom he...

Headers: +

ACL: +

Web acceleration: Compression Cache HTTP->HTTPS

WAF: Slow attack DDOS Whitelist Blacklist WAF Antibot

Options: Forward for Redispatch Force HTTP close Set cookie Set options

localhost : 5000 Port check: 5000 maxconn: 200 send-proxy backup

Figure 20: Configure proxy settings

After filling in the details click on *Add* to add the configuration to the HAProxy server.

**Overview of the fields in the configuration:**

<b>FIELD</b>	<b>FUNCTION</b>
<b>SELECT A SERVER:</b>	The server that the proxy will be configured on.
<b>NAME:</b>	Name of the proxy connection.
<b>IP AND PORT:</b>	The public port on the server that will be listening.
<b>MODE:</b>	The type of proxy, TCP is used here.
<b>SERVERS:</b>	The IP and port the packet will be forwarded to.

After adding the configuration given above and restarting the HAProxy service, packets from port 8080 on the public interface will be forwarded to local port 5000 on the server, which the Windows client will bind to.

## 6 Extras

### 6.1 How to create a new SSH key?

1. Open a shell on the local machine and create a new keypair.

```
ssh-keygen -t ed25519 -f .ssh\haproxy_key -C "haproxy key"
```

2. Using the following command, the key can be read:

```
cat .ssh\haproxy_key.pub
```

## 7 Conclusion

In conclusion, organizations looking to strike a balance between security and accessibility can use HAProxy as a reverse proxy to give outside access to services operating behind NAT or firewalls.

Furthermore, by centralizing access, IT teams can more easily apply security guidelines, monitor statuses of services.

Considering these advantages, it may be worthwhile for businesses to explore the use of HAProxy as a reverse proxy, which can enable external users to access services operating behind NAT or firewalled devices. To guarantee the best security and performance, it is crucial to note that the reverse proxy must be properly configured and managed. The effectiveness of HAProxy in various organizational contexts and configurations, as well as alternative approaches to enabling external access to internal services while keeping security and control, may be explored in further study.



## 8 Source references

JFrog. (2021, August 15). Reverse SSH Tunnelling: From Start to End.

<https://jfrog.com/connect/post/reverse-ssh-tunneling-from-start-to-end/>

Dan Nanni. (2020, November 21). Access Linux Server Behind NAT via Reverse SSH Tunnel.

<https://www.xmodulo.com/access-linux-server-behind-nat-reverse-ssh-tunnel.html>

HAProxy Technologies. (2021). Configuration Manual.

<https://www.haproxy.com/documentation/>

Roxy-WI. (n.d.).

<https://roxy-wi.org/installation>

Cisco. (2020). What Is Network Address Translation?

<https://www.cisco.com/c/en/us/support/docs/ip/network-address-translation-nat/26704-nat-faq-00.html>

MDN Web Docs. (2023). Proxy servers and tunnelling.

[https://developer.mozilla.org/en-US/docs/Web/HTTP/Proxy\\_servers\\_and\\_tunneling](https://developer.mozilla.org/en-US/docs/Web/HTTP/Proxy_servers_and_tunneling)

HAProxy. (2023). The Reliable, High-Performance TCP/HTTP Load Balancer.

<http://www.haproxy.org/>

Roxy-WI. (2023). Roxy-WI server management.

<https://roxy-wi.org/>

Microsoft. (2023). Task Scheduler.

<https://docs.microsoft.com/en-us/windows/win32/taskschd/task-scheduler-start-page>

Cisco Umbrella. (2023, February 23). What is a Proxy Server?

<https://umbrella.cisco.com/blog/what-is-a-proxy-server>

DigitalOcean. (2022, March 28). HAProxy

<https://www.digitalocean.com/community/tutorials/an-introduction-to-haproxy-and-load-balancing-concepts>