

Firewall

- [Installation d'un pare-feu basique avec iptables](#)
- [Firewalld : un firewall simple à utiliser](#)
- [UFW - Basic Setup](#)
- [UFW - Block entire countries by IPs](#)

Installation d'un pare-feu basique avec iptables

Installation de iptables

```
apt install iptables
```

Création du process et des règles

Je créer un fichier firewall qui va être exécuter au démarrage et gérer par systemctl.

Son but est d'indiquer quoi faire quand on start le process et quand on le stop et la gestion du status.

```
nano /etc/init.d/firewall
```

```
#!/bin/sh

### BEGIN INIT INFO
# Provides:      Firewall
# Required-Start: $remote_fs $syslog
# Required-Stop:  $remote_fs $syslog
# Default-Start:  2 3 4 5
# Default-Stop:   0 1 6
# Short-Description: Firewall
# Description:    Firewall
### END INIT INFO

IPT=/sbin/iptables
IP6T=/sbin/ip6tables

do_start() {
    # Efface toutes les regles en cours. -F toutes. -X utilisateurs
    $IPT -t filter -F
    $IPT -t filter -X
```

```

$IPT -t nat -F
$IPT -t nat -X
$IPT -t mangle -F
$IPT -t mangle -X
$IP6T -t filter -F
$IP6T -t filter -X
$IP6T -t mangle -F
$IP6T -t mangle -X

# strategie (-P) par default : bloc tout l'entrant le forward et autorise le sortant
$IPT -t filter -P INPUT DROP
$IPT -t filter -P FORWARD DROP
$IPT -t filter -P OUTPUT ACCEPT
$IP6T -t filter -P INPUT DROP
$IP6T -t filter -P FORWARD DROP
$IP6T -t filter -P OUTPUT ACCEPT

# Loopback
$IPT -t filter -A INPUT -i lo -j ACCEPT
$IPT -t filter -A OUTPUT -o lo -j ACCEPT
$IP6T -t filter -A INPUT -i lo -j ACCEPT
$IP6T -t filter -A OUTPUT -o lo -j ACCEPT

# Permettre a une connexion ouverte de recevoir du trafic en entree
$IPT -t filter -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
$IP6T -t filter -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT


echo "firewall started [OK]"
}

# fonction qui arrete le firewall
do_stop() {
    # Efface toutes les regles
    $IPT -t filter -F
    $IPT -t filter -X
    $IPT -t nat -F
    $IPT -t nat -X
    $IPT -t mangle -F
    $IPT -t mangle -X
    $IP6T -t filter -F
    $IP6T -t filter -X
    $IP6T -t mangle -F
    $IP6T -t mangle -X

    # remet la strategie
    $IPT -t filter -P INPUT ACCEPT

```

```

$IPT -t filter -P OUTPUT ACCEPT
$IPT -t filter -P FORWARD ACCEPT
$IP6T -t filter -P INPUT ACCEPT
$IP6T -t filter -P OUTPUT ACCEPT
$IP6T -t filter -P FORWARD ACCEPT
#
echo "firewall stopped [OK]"
}
# fonction status firewall
do_status() {
    # affiche les regles en cours
    clear
    echo Status IPV4
    echo -----
    $IPT -L -n -v
    echo
    echo -----
    echo
    echo status IPV6
    echo -----
    $IP6T -L -n -v
    echo
}
case "$1" in
    start)
        do_start
        exit 0
    ;;
    stop)
        do_stop
        exit 0
    ;;
    restart)
        do_stop
        do_start
        exit 0
    ;;
    status)
        do_status
        exit 0
    ;;

```

```

*)
    echo "Usage: /etc/init.d/firewall {start|stop|restart|status}"
    exit 1
;;
esac

```

Nous avons ici la configuration la plus bloquante (trafic sortant uniquement): Uniquement le trafic local et les connexion établie vers l'extérieur en IPv4 et IPv6.

Pour ajouter des autorisation, par exemple l'ICMP (ping), dans `do_start()` ajouter:

```

❏# ICMP
$IPT -t filter -A INPUT -p icmp -j ACCEPT
$IP6T -t filter -A INPUT -p ipv6-icmp -j ACCEPT

```

Votre `do_start()` ressemble à ça:

```

do_start() {
    # Efface toutes les regles en cours. -F toutes. -X utilisateurs
    $IPT -t filter -F
    $IPT -t filter -X
    $IPT -t nat -F
    $IPT -t nat -X
    $IPT -t mangle -F
    $IPT -t mangle -X
    $IP6T -t filter -F
    $IP6T -t filter -X
    $IP6T -t mangle -F
    $IP6T -t mangle -X

    # strategie (-P) par default : bloc tout l'entrant le forward et autorise le sortant
    $IPT -t filter -P INPUT DROP
    $IPT -t filter -P FORWARD DROP
    $IPT -t filter -P OUTPUT ACCEPT
    $IP6T -t filter -P INPUT DROP
    $IP6T -t filter -P FORWARD DROP
    $IP6T -t filter -P OUTPUT ACCEPT

    # Loopback
    $IPT -t filter -A INPUT -i lo -j ACCEPT
    $IPT -t filter -A OUTPUT -o lo -j ACCEPT
    $IP6T -t filter -A INPUT -i lo -j ACCEPT
    $IP6T -t filter -A OUTPUT -o lo -j ACCEPT

    # Permettre a une connexion ouverte de recevoir du trafic en entree

```

```
$IPT -t filter -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
$IP6T -t filter -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT

# ICMP

$IPT -t filter -A INPUT -p icmp -j ACCEPT
$IP6T -t filter -A INPUT -p ipv6-icmp -j ACCEPT

echo "firewall started [OK]"

}
```

Voici une liste de règles souvent utile

```
# ICMP

$IPT -t filter -A INPUT -p icmp -j ACCEPT
$IP6T -t filter -A INPUT -p ipv6-icmp -j ACCEPT

# DNS:53

$IPT -t filter -A INPUT -p tcp --dport 53 -j ACCEPT
$IPT -t filter -A INPUT -p udp --dport 53 -j ACCEPT
$IP6T -t filter -A INPUT -p tcp --dport 53 -j ACCEPT
$IP6T -t filter -A INPUT -p udp --dport 53 -j ACCEPT

#FTP:20+21 - FTP pasv: 10 090 - 10 100
$IPT -t filter -A INPUT -p tcp --dport 20 -j ACCEPT
$IP6T -t filter -A INPUT -p tcp --dport 20 -j ACCEPT
$IPT -t filter -A INPUT -p tcp --dport 21 -j ACCEPT
$IP6T -t filter -A INPUT -p tcp --dport 21 -j ACCEPT
$IPT -t filter -A INPUT -p tcp --dport 10090:10100 -j ACCEPT
$IP6T -t filter -A INPUT -p tcp --dport 10090:10100 -j ACCEPT

# SSH:22

# ATTENTION, indiques bien ton port personnalisé si tu l'as changé
$IPT -t filter -A INPUT -p tcp --dport 22 -j ACCEPT
$IP6T -t filter -A INPUT -p tcp --dport 22 -j ACCEPT

# HTTP:80 (Serveur Web)
$IPT -t filter -A INPUT -p tcp --dport 80 -j ACCEPT
$IP6T -t filter -A INPUT -p tcp --dport 80 -j ACCEPT

# HTTPS:443 (Serveur Web)
$IPT -t filter -A INPUT -p tcp --dport 443 -j ACCEPT
```

```
$IP6T -t filter -A INPUT -p tcp --dport 443 -j ACCEPT
```

```
# SNMP:161 (Monitoring)
```

```
$IPT -t filter -A INPUT -p udp --dport 161 -j ACCEPT
```

```
$IP6T -t filter -A INPUT -p udp --dport 161 -j ACCEPT
```

```
# SMTP:25 (Mail)
```

```
❏$IPT -t filter -A INPUT -p tcp --dport 25 -j ACCEPT
```

```
❏$IP6T -t filter -A INPUT -p tcp --dport 25 -j ACCEPT
```

```
# POP3:110 (Mail)
```

```
❏$IPT -t filter -A INPUT -p tcp --dport 110 -j ACCEPT
```

```
❏$IP6T -t filter -A INPUT -p tcp --dport 110 -j ACCEPT
```

```
❏# IMAP:143 (Mail)
```

```
❏$IPT -t filter -A INPUT -p tcp --dport 143 -j ACCEPT
```

```
❏$IP6T -t filter -A INPUT -p tcp --dport 143 -j ACCEPT
```

Il faut maintenant rendre exécutable de fichier

```
chmod +x /etc/init.d/firewall
```

Les prochaines étapes vont rendre actif le pare-feu. Si vous êtes en SSH et que une règle n'autorise pas le SSH vous risquer de vous auto bloqué a la prochaine connexion. Assurez-vous d'avoir un accès directe à la machine pour gérer le cas ou vous seriez dans l'incapacité de vous reconnecter.

Premier démarrage du pare-feu

```
/etc/init.d/firewall start
```

Gestion par systemctl

```
update-rc.d firewall defaults
```

```
systemctl enable firewall
```

```
systemctl start firewall
```

```
systemctl status firewall
```

Cela devrais vous indiquer quelque chose du genre:

● firewall.service - LSB: Firewall

Loaded: loaded (/etc/init.d/firewall; generated)

Active: active (exited) since *** ***_**_** **:**:** ***: * ago

Docs: man:systemd-sysv-generator(8)

Process: ***** ExecStart=/etc/init.d/firewall start (code=exited, status=0/SUCCESS)

***. ** **:**:** monitor systemd[1]: Starting LSB: Firewall...

***. ** **:**:** monitor firewall[30462]: firewall started [OK]

***. ** **:**:** monitor systemd[1]: Started LSB: Firewall.

Firewalld : un firewall simple à utiliser

Firewalld est un pare-feu que je trouve très agréable à utiliser, où on peut « cacher » la complexité de certains éléments de configuration derrière des noms simples à utiliser.

Par exemple, je peux avoir un service qui n'a pas spécialement de port dédié, donc qui n'est pas proposé par firewall. Mettons un wireguard qui écoute sur le port 9879. Plutôt que d'utiliser `9879/udp` dans ma configuration, je vais créer un service `wireguard`, et c'est ce service que j'autoriserai.

Ce sera bien plus parlant quand je relirai la configuration.

Liens :

- Documentation officielle : <https://firewalld.org/documentation/>
- <https://www.linuxtricks.fr/wiki/firewalld-le-pare-feu-facile-sous-linux>
- <https://www.rootusers.com/how-to-use-firewalld-rich-rules-and-zones-for-filtering-and-nat/>
- <https://kb.vander.host/security/firewalld-cheat-sheet/>

Principes

En très gros et en très résumé, on va avoir des zones, qui sont des ensembles d'adresses IP et la zone `public` qui concerne toutes les IPs, sauf celles qui sont dans d'autres zones.

On va aussi avoir des services, qui décrivent... des services : port et protocole (ex: `5666` et `tcp` pour nrpe).

On va aussi avoir des « rich rules » dans les zones, des exceptions aux règles appliquées dans la zone.

Toute la configuration est dans des fichiers XML très simples à lire, c'est très agréable. Tant qu'on ne surcharge pas la configuration par défaut, les fichiers sont dans `/usr/lib/firewalld/` mais dès qu'on modifie un élément de configuration, celui-ci se retrouvera copié dans `/etc/firewalld/` et modifié.

Test de configuration

Si on modifie de la configuration à la main (en écrivant dans `/etc/firewalld`, on prendra soin de tester la configuration avec les commandes suivantes :

Si `firewalld` est coupé :

```
firewall-offline-cmd --check-config
```

Si `firewalld` est lancé :

```
firewall-cmd --check-config
```

Attention

Quand on installe `firewalld`, le firewall démarre de suite en n'autorisant en public que `ssh` (et `dhcpv6-client`).

Il est donc préférable de l'installer et de le couper directement après :

```
apt install firewalld &&  
systemctl stop firewalld
```

On pourra créer la configuration tranquillement avec la commande `firewall-offline-cmd` et lancer le service une fois la configuration terminée.

Changements permanents

Si on veut rendre un changement permanent (c-à-d qu'il soit écrit dans la config au lieu d'être juste appliqué jusqu'au redémarrage), il faut ajouter ça aux commandes :

```
--permanent
```

Par contre, avec `--permanent`, il faut recharger la configuration pour appliquer les modifications (ou alors on applique une fois avec `--permanent` et une fois sans) :

```
firewall-cmd --reload
```

À l'inverse, on peut créer des règles sans le `--permanent` et ensuite écrire ces règles dans la configuration permanent avec la commande suivante :

```
firewall-cmd --runtime-to-permanent
```

NB : certaines commandes nécessitent forcément le `--permanent`.

Bloquer une adresse IP

On peut soit ajouter les adresses aux zones `drop` ou `block` :

```
firewall-cmd --zone=drop --add-source 192.0.2.0/24
firewall-cmd --zone=drop --add-source 192.0.2.0/24 --permanent
```

Soit ajouter une `rich-rule` (`man firewalld.richlanguage`) à la zone `public` :

```
firewall-cmd --zone public --add-rich-rule "rule family=ipv4 source address=192.0.2.0/24 reject"
firewall-cmd --zone public --add-rich-rule "rule family=ipv4 source address=192.0.2.0/24 reject" --permanent
```

Pour enlever un blocage :

```
firewall-cmd --zone drop --remove-source 51.159.0.0/16
firewall-cmd --zone drop --remove-source 51.159.0.0/16 --permanent
```

```
firewall-cmd --zone public --remove-rich-rule "rule family=ipv4 source address=51.159.0.0/16 reject"
firewall-cmd --zone public --remove-rich-rule "rule family=ipv4 source address=51.159.0.0/16 reject" --
permanent
```

Pour voir les blocages par `rich-rule` (la 1ère commande donne les blocages actuellement activés, l'autre ceux qui sont dans les fichiers de configuration. Il peut y avoir une différence... ou pas !) :

```
firewall-cmd --list-rich-rules
firewall-cmd --list-rich-rules --permanent
```

Pour bloquer un ipset (voir plus bas) :

```
firewall-cmd --zone=drop --add-source ipset:le_nom_de_l_ipset
firewall-cmd --zone=drop --add-source ipset:le_nom_de_l_ipset --permanent
```

Zones

Voir les zones disponibles :

```
firewall-cmd --get-zones
```

NB : la zone `public`, par défaut, n'autorise que le SSH et dhcpv6-client. L'installation de firewalld sur une machine va donc couper l'accès aux services. Il faut donc stopper firewalld juste après son installation, regarder les ports utilisés sur la machine et modifier la zone `public` soit en copiant `/usr/lib/firewalld/zones/public.xml` dans `/etc/firewalld/zones/`, soit en préparant une ligne de commande à lancer juste après le démarrage de firewalld.

NB : les zones peuvent avoir une `target`, l'action à appliquer aux connexions qui correspondent à la zone. Voir [la doc](#).

NB : Une adresse IP ne peut se trouver que dans une seule zone mais on peut ajouter dans une zone un réseau qui contient une adresse IP déjà présente dans une autre zone. Cependant, le comportement peut ne pas être celui attendu. Il vaut mieux ajouter une `rich-rule` à la zone pour faire une exception aux règles de la zone.

Voir la zone par défaut (celle sur laquelle s'appliqueront les modifications si on ne spécifie pas la zone) :

```
firewall-cmd --get-default-zone
```

Définir la zone par défaut :

```
firewall-cmd --set-default-zone work
```

Voir la configuration de la zone :

```
firewall-cmd --info-zone lazone
```

Voir la configuration de toutes les zones :

```
firewall-cmd --list-all-zones
```

Créer une zone :

```
firewall-cmd --permanent --new-zone mazonne  
firewall-cmd --reload
```

Supprimer une zone :

```
firewall-cmd --permanent --delete-zone mazon
```

```
firewall-cmd --reload
```

Chaque interface du système peut être attribuée à une zone. Pour ajouter l'interface ens192 à la zone work en l'enlevant de sa précédente zone :

```
firewall-cmd --change-interface ens192 --zone work [--permanent]
```

Pour retirer l'interface ens192 de la zone work :

```
firewall-cmd --remove-interface ens192 --zone work [--permanent]
```

Pour ajouter l'interface ens192 à la zone work (interface qui ne soit pas être affectée à une zone) :

```
firewall-cmd --add-interface ens192 --zone work [--permanent]
```

Ajouter des adresses IP ou un réseau à une zone :

```
firewall-cmd --zone work --add-source 192.0.2.0/24 [--permanent]
```

```
firewall-cmd --zone work --add-source 192.0.2.200 [--permanent]
```

Retirer des adresses IP ou un réseau d'une zone :

```
firewall-cmd --zone work --remove-source 192.0.2.0/24 [--permanent]
```

```
firewall-cmd --zone work --remove-source 192.0.2.200 [--permanent]
```

Pour basculer une adresse IP ou un réseau d'une zone à une autre :

```
firewall-cmd --zone l_autre_zone --change-source 192.0.2.0/24 [--permanent]
```

```
firewall-cmd --zone l_autre_zone --change-source 192.0.2.200 [--permanent]
```

Si l'adresse IP / le réseau était dans une autre zone, ça équivaut à un `--remove-source` suivi d'un `--add-source`, si ce n'était pas le cas, ça fait juste comme un `--add-source`.

Voir la `target` d'une zone :

```
firewall-cmd --permanent --get-target --zone drop
```

Définir la `target` d'une zone :

```
firewall-cmd --permanent --set-target [default|ACCEPT|DROP|REJECT] --zone drop
```

Pour voir dans quelle zone est une adresse IP :

```
firewall-cmd --get-zone-of-source=<adresse IP ou réseau en notation CIDR ou adresse MAC ou ipset>
```

Si ça répond `no zone`, c'est que l'IP ou le réseau n'est pas explicitement associé à une zone.

Attention : si un réseau est enregistré dans une zone, lancer la commande sur une IP du réseau ne renverra pas la zone en question !

Services

Voir les services existants :

```
firewall-cmd --get-services
```

Voir le détail d'un service :

```
firewall-cmd --info-service ssh
```

Créer un nouveau service :

```
firewall-cmd --permanent --new-service influxdb  
firewall-cmd --permanent --service influxdb --set-description InfluxDB  
firewall-cmd --permanent --service influxdb --add-port 8086/tcp
```

Ajouter un service à une zone :

```
firewall-cmd --zone public --add-service nrpe [--permanent]
```

Retirer un service d'une zone :

```
firewall-cmd --zone public --remove-service nrpe [--permanent]
```

Voir les services d'une zone :

```
firewall-cmd --list-services --zone work
```

Si on ne souhaite pas créer de service mais autoriser un certain port et protocole, on peut les ajouter directement à la zone :

```
firewall-cmd --zone work --add-port 1234/udp [--permanent]
```

Et pour les supprimer :

```
firewall-cmd --zone work --remove-port 1234/udp [--permanent]
```

Pour voir les ports/protocoles d'une zone (ça ne listera pas les services !) :

```
firewall-cmd --list-ports --zone work
```

IPSet : groupes d'adresses

Doc : https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/7/html/security_guide/sec-setting_and_controlling_ip_sets_using_firewalld

Cas d'usage : on voit plein de spammeurs venant du VPN d'Avast. On fait un `whois`, on voit le numéro d'AS des serveurs d'Avast, on va sur <https://asnlookup.com/> pour choper leur liste d'adresses IP et on en fait un ipset pour les bloquer.

NB : un ipset ne peut contenir qu'un type d'adresses, IPv4 ou IPv6, pas les deux.

NB : Fail2ban, lorsqu'il utilise firewalld, utilise des ipset, mais à un niveau un peu plus bas (nftables). Ces ipset ne sont pas vus par firewalld. On peut voir tous les ipset, même bas niveau avec la commande `ipset list -name` (`ipset list le_nom_de_l_ipset` pour voir les adresses et le détail de l'ipset).

Un ipset sert à regrouper des adresses pour leur appliquer des règles facilement.

Voir les types d'ipset disponibles :

```
firewall-cmd --get-ipset-types
```

Voir les ipsets existants :

```
firewall-cmd --permanent --get-ipsets
```

Créer un ipset :

```
firewall-cmd --permanent --type hash:net --new-ipset test
```

Pour un ipset IPv6 :

```
firewall-cmd --permanent --type hash:net --option "family=inet6" --new-ipset test-v6
```

Supprimer un ipset :

```
firewall-cmd --permanent --delete-ipset test
```

Voir les infos d'un ipset :

```
firewall-cmd --info-ipset test [--permanent]
```

Ajouter une adresse IP à un ipset :

```
firewall-cmd --ipset test --add-entry 192.0.2.1 [--permanent]
```

Supprimer une adresse IP d'un ipset :

```
firewall-cmd --ipset test --remove-entry 192.0.2.1 [--permanent]
```

Voir les adresses IP d'un ipset :

```
firewall-cmd --ipset test --get-entries [--permanent]
```

Ajouter un paquet d'IP d'après un fichier :

```
cat > iplist.txt <<EOL
192.0.2.2
192.0.2.3
198.51.100.0/24
203.0.113.254
EOL
firewall-cmd --ipset test --add-entries-from-file iplist.txt [--permanent]
```

Supprimer un paquet d'IP d'après un fichier :

```
firewall-cmd --ipset test --remove-entries-from-file iplist.txt [--permanent]
```

Ajouter un ipset dans une zone :

```
firewall-cmd --zone drop --add-source ipset:test [--permanent]
```

Supprimer un ipset d'une zone :

```
firewall-cmd --zone drop --remove-source ipset:test [--permanent]
```


Créer des exceptions avec des règles riches

Cas classique : on bloque un pays avec des ipset et quelqu'un a besoin d'accéder à nos services depuis là-bas.

L'ipset est dans la zone `drop`. On va ajouter une `rich-rule` pour faire une exception :

```
firewall-cmd --zone drop --add-rich-rule='rule family="ipv4" source address="192.0.2.29" port port="443" protocol="tcp" accept'
firewall-cmd --zone drop --add-rich-rule='rule family="ipv4" source address="192.0.2.29" port port="80" protocol="tcp" accept'
firewall-cmd --zone drop --permanent --add-rich-rule='rule family="ipv4" source address="192.0.2.29" port port="443" protocol="tcp" accept'
firewall-cmd --zone drop --permanent --add-rich-rule='rule family="ipv4" source address="192.0.2.29" port port="80" protocol="tcp" accept'
```

Pour supprimer l'exception :

```
firewall-cmd --zone drop --remove-rich-rule 'rule family="ipv4" source address="192.0.2.29" port port="80" protocol="tcp" accept'
firewall-cmd --zone drop --remove-rich-rule 'rule family="ipv4" source address="192.0.2.29" port port="443" protocol="tcp" accept'
firewall-cmd --zone drop --permanent --remove-rich-rule 'rule family="ipv4" source address="192.0.2.29" port port="80" protocol="tcp" accept'
firewall-cmd --zone drop --permanent --remove-rich-rule 'rule family="ipv4" source address="192.0.2.29" port port="443" protocol="tcp" accept'
```

On peut utiliser des services dans les règles riches :

```
firewall-cmd --zone drop --add-rich-rule='rule family="ipv4" source address="192.0.2.29" service name=https accept'
```

On peut utiliser des ipset dans les règles riches :

```
firewall-cmd --zone drop --add-rich-rule='rule family="ipv4" source ipset="test-v6" service name=https accept'
```

NB : ajouter l'adresse IP en source dans la zone `public` ne servirait strictement à rien.

Blocage geoIP

On peut se baser sur le script de <https://github.com/simonbouchard/geoip-blocking-w-firewalld> (le fork de Framasoft).

On modifie les pays à bloquer dans `/etc/default/firewalld-geoip` et on lance le script. À mettre dans un cron pour mettre à jour les adresses.

Faire du NAT

Voir la partie `Network Address Translation (NAT)` de <https://www.rootusers.com/how-to-use-firewalld-rich-rules-and-zones-for-filtering-and-nat/>.

UFW - Basic Setup

1. Installation and Configuration

First, install UFW

```
sudo apt -y install ufw
```

Before enabling the setup, we will set up some basic rules. I will deny all outgoing, as well as all incoming traffic as a default. After that we have to make sure, that we enable all the necessary protocols to communicate, otherwise, basic services, like DNS resolution no longer work. UFW is basically just a script, that generates IP-Table entries for you.

Disable all outgoing and incoming traffic:

```
sudo ufw default deny incoming  
sudo ufw default deny outgoing
```

Now enable logging

```
sudo ufw logging FULL
```

Next, you have to decide, which outgoing traffic to allow. Here is an overview of **some** services, and which default ports and protocols they use. This overview is only for **OUTGOING** traffic.

Service	Port	Protocol
SMTP	25	tcp/udp
SMTPs	465	tcp/udp
DNS	53	tcp/udp
HTTP	80	tcp (UDP usually not needed)
HTTPS	443	tcp (UDP usually not needed)

You can enable outgoing traffic like this:

```
sudo ufw allow out PORT/Protocol
```

So to enable DNS, run

```
sudo ufw allow out 53
```

To enable ICMP, you'll have to edit the IP-Tables yourself, since UFW doesn't offer you this feature. Just add the following lines to `/etc/ufw/before.rules` and `/etc/ufw/before6.rules`

```
-A ufw-before-output -p icmp -m state --state NEW,ESTABLISHED,RELATED -j ACCEPT
-A ufw-before-output -p icmp -m state --state ESTABLISHED,RELATED -j ACCEPT
```

After that, you can enable incoming traffic. On a freshly installed system, that's usually just SSH, but if you are running, e.g. a webserver, you should also enable traffic on Ports 80 and 443. UFW comes with an App-List, which makes adding UFW rules for default ports easier. You can view the list by running

```
sudo ufw app list
```

Allow incoming traffic like this:

```
sudo ufw allow in Port/Protocol
```

If you want to allow a range of ports, you can use `PORTX:PORTY`. If you want only a certain host to be able to connect to your client via some Port, you can use the following rule:

```
sudo ufw allow in from 10.10.10.10 to any port 22
```

You can also use netmasks in CIDR format

You can do much more. I recommend you to read through this [manpage](#).

Lastly, to enable UFW, run

```
sudo ufw enable
```

2. Tips and Tricks

To quickly delete all rules for a port, you can use the following script:

```
#!/bin/bash
for NUM in $(ufw status numbered | grep "$1" | awk -F"[]" '{print $2}' | tr --delete "[:blank:]" | sort -rn); do
    ufw --force delete "$NUM"
done
```

This will instantly delete all rules for the specified ports, without any prompts.

You can simply delete all rules for e.g. Port 80, by running.

```
sudo ./your-script-name.sh 80
```

UFW - Block entire countries by IPs

If you don't want one explicit or even multiple countries, to connect to your server, you can block all their requests, by using UFW. This will, however, make your UFW status output extremely long, as well as taking some time to set up.

Doing this, is, of course, no guarantee, because it only blocks IPv4 and often enough, attackers mask/spoof their IP anyway. Moreover, this is quite a drastic step to take. You should rather control access on a different level, e.g. using a WAF.

First, you need to download a list of all the IPv4 Subnets, a country has assigned. You can download said list [here](#) and make sure to choose CIDR as output format. After that, simply run the following command (This may take a while)

```
cat yourList.txt | awk '/^[^#]/ { print $1 }' | sudo xargs -l {} ufw deny from {} to any
```

To remove all the rules, run

```
cat yourList.txt | awk '/^[^#]/ { print $1 }' | sudo xargs -l {} ufw delete deny from {}
```