SPOC Tunnel BASH/ZSH Function

This document includes two main sections.

1. The steps needed to run `sshuttle` with domain-specific-dns (which forwards only requests for spoc.charterlab.com to the spoc-jumphost).
2. A function that can be added to your zsh/bash profile that allows a simple, one-step fingerprint authentication for the `sshuttle` command. This is accomplished by using the 1Password CLI utiliity "`op`" to read the appropriate passwords and securely pass them to the `sshuttle` utlity. I have marked the steps for this method as "OPTIONAL".

Requirements

Homebrew (MacOS) package manager

- Homebrew Installation Instructions

SSHuttle

```
brew install sshuttle
```

SSHPass (Optional)

This utility allows you to forward a password to the `ssh` command. This example uses an environment variable that in turn uses a password manager to securely pipe password information to `sshuttle`.

WARNING: There are methods to invoke `sshpass` that **ARE NOT** secure. As such, the utility is not available directly from homebrew. I use a custom tap and formula hosted on my personal github account.

```
brew tap ajanis/custombrew
brew install ajanis/custombrew/sshpass
```

1Password and CLI utlity (Optional)

This is for 1-touch authentication. 1Password is approved for use with an enterprise licence (which I have) -- However I will be investigating additional options using both the Mac OS Keychain and Keepass for increased compatibility/compliance)

```
brew install --cask 1password 1password-cli
```

Sudo privileges

You will need to modify /etc/sudoers if you have not already so you can run commands as a privileged user.

The configuration below is OPTIONAL:

```
# Cmnd alias specification
Cmnd_Alias CUSTOM = /usr/local/bin/sshuttle, /usr/bin/pgrep,
/usr/bin/pkill, /usr/sbin/visudo, /usr/local/bin/sshpass,
/usr/local/bin/op

# User specification
    # root and users in group wheel can run anything on any machine as any
user
root        ALL = (ALL) ALL
    # users in group admin can run any command from any machine as any
user
    # users from group admin can also run any command from the CUSTOM
alias on any machine as any user without a password
%admin      ALL = (ALL) ALL : ALL = (ALL) NOPASSWD: CUSTOM
```

Modify your sshuttle command to ONLY send DNS requests for 'spoc.charterlab.com' to the sshuttle jumphost

Create a domain-specific-DNS configuration for the SPOC Lab

(Credit to Josh Hurtado for these instructions)

- Run the following commands as a priviliged / root user on the MacOS machine:
- Create the directory /etc/resolver

```
sudo mkdir /etc/resolver
```

- Create the resolver file for SPOC at /etc/resolver/spoc.charterlab.com

```
sudo echo 'search spoc.charterlab.com spoc.local nameserver 172.22.73.19'
> /etc/resolver/spoc.charterlab.com
```

- Verify the new resolver for spoc.charterlab.com is present. *(You will have to scroll down a bit to find the correct one)*

```
❯ sudo scutil --dns
DNS configuration
...
resolver #8
```

```
    domain   : spoc.charterlab.com
    search domain[0] : spoc.charterlab.com
    search domain[1] : spoc.local
    search domain[2] : nameserver
    search domain[3] : 172.22.73.19
    flags    : Request A records, Request AAAA records
    reach    : 0x00000000 (Not Reachable)
```

Create files for `allow` and `deny` that will be used by `sshuttle`

The `--dns` flag sends *all* requests to the **sshuttle jumphost**. You will need to modify your `sshuttle` command so that *ONLY* requests for `spoc.charterlab.com` will be handled by the **sshuttle jumphost**.

- ALLOW file command

```
echo << EOF >> ~/.spoc.allow.txt
44.0.0.0/8
10.240.12.0/22
10.244.28.0/22
10.240.40.0/22
10.240.64.0/23
10.240.72.0/22
10.240.76.0/22
#Optical's polatis
10.252.254.197/32
10.252.254.9/24
10.252.255.0/24
172.22.32.0/24
172.22.73.31/32
172.22.73.70
172.22.73.99
172.22.73.19
172.22.73.27/32
172.22.73.164/32
#172.22.73.0/24
#172.22.72.0/22
172.23.62.0/24
172.30.124.128/26
172.22.73.128/25
172.22.73.126
172.22.73.127
172.23.35.32/27
35.135.193.64/26
35.135.193.0/24
2600:6ce6:4410::/48
2605:1c00:50f2::/48
2600:6ce7:0:5::/64
2600:6cec:1c0:7::/64
#2605:1c00:50f2:2800::/64
2605:1c00:50f2:280e::/64
2605:1c00:50f2:280e::6100/64
```

```
2605:1c00:50f2:2800:172:22:73:100/128
2605:1c00:50f2:2800:172:22:73:164/128
2605:1c00:50f2:2800:172:22:73:31/128
EOF
```

- DENY File command

```
echo << EOF >> ~/.spoc.deny.txt
#corp
142.136.0.0/16
142.136.235.173
22.0.0.0/8
33.0.0.0/8
#10.151.0.0/16
#SPOC
#10.240.72.137
35.135.192.78/32
#172.23.62.20
172.23.62.21
172.22.73.17
#172.22.73.13
#2605:1c00:50f2:2800:172:22:73:17/128
#2605:1c00:50f2:2800:172:22:73:13/128
#2605:1c00:50f2:2800:172:22:73:18/128
#2605:1c00:50f2:280b:172:23:62:222/128
EOF
```

Run the `sshuttle` command in your terminal

```
sshuttle -v -r $USER@35.135.192.78:3022 -s ~/.spoc.allow.txt -X
~/.spoc.deny.txt --ns-hosts 172.22.73.19 --to-ns 172.22.73.19
```

Stop here if you do not wish to create the `helper function` or use the `1-touch authentication` process

---

Create the function that we will use to start and stop the `sshuttle` process (OPTIONAL)

This very simple function is just a wrapper for the `sshuttle` command that incorporates the 1-touch/1Password features.

You can modify the function to invoke `sshuttle` with any arguments you wish. And of course, you can still run the `sshuttle` command from the CLI as always.

```
echo << EOF >> ~/.spoc.rc
spoctunnel () {
SSHUTTLESTATE=$1
LOGFILE="$HOME/.sshuttle.log"

case $SSHUTTLESTATE in
    start)
      if ! pgrep -f sshuttle; then
      echo > $LOGFILE
      SSHPASS=$(op read op://Charter/charterlab-spoc/password) \
      bash -c 'sshpass -e sshuttle -v -r $USER@35.135.192.78:3022 \
      -s .spoc.allow.txt \
      -X .spoc.deny.txt \
      --ns-hosts 172.22.73.19 \
      --to-ns 172.22.73.19' >>$LOGFILE 2>&1 &
      fi
      ;;
    origstart)
      if ! pgrep -f sshuttle; then
      echo > $LOGFILE
      SSHPASS=$(op read op://Charter/charterlab-spoc/password) \
      bash -c 'sshpass -e sshuttle --dns -v -r $USER@35.135.192.78:3022 \
      10.240.41.0/22 \
      10.240.12.0/22 \
      44.128.28.0/24 \
      44.114.100.0/24 \
      44.128.32.0/22 \
      44.128.124.0/22 \
      172.22.73.0/24 \
      10.240.72.0/22 \
      44.130.0.0/16 \
      172.23.35.32/27 \
      172.30.105.0/16 \
      44.140.0.0/16 \
      44.130.8.0/16 \
      35.135.193.123/26 \
      44.130.8.0/22 \
      44.114.1.142/24' >>$LOGFILE 2>&1 &
      fi
      ;;
    stop)
      if pgrep -f sshuttle; then
      sudo pkill -f sshuttle >>$LOGFILE 2>&1
      fi
      ;;
    tail)
      tail -F $LOGFILE
      ;;
    cat)
      cat $LOGFILE
      ;;
    *)
      echo -e "$0 (start|stop|tail|cat|origstart)
      start:      Starts sshuttle using -s ~/.spoc.allow.txt and -X
~/.spoc.deny.txt
      stop:       Shuts down the sshuttle application
      tail:       Tails the sshuttle process log file at ~/.sshuttle.log
      cat:        Displays the entire file at ~/.sshuttle.log
      origstart:  Starts the sshuttle application using the --dns flag and
the original list of subnets"
      ;;
esac
}
EOF
```

Include ~/.spoc.rc you just created in your $SHELL profile / rcfile (OPTIONAL)

This example uses .zshrc, but you can substitute the rcfile for your $SHELL of choice

```
cat << EOF >> .zshrc
[[ -f ~/.spoc.zsh ]] && source ~/.spoc.zsh
EOF
```

Run sshuttle helper (OPTIONAL)

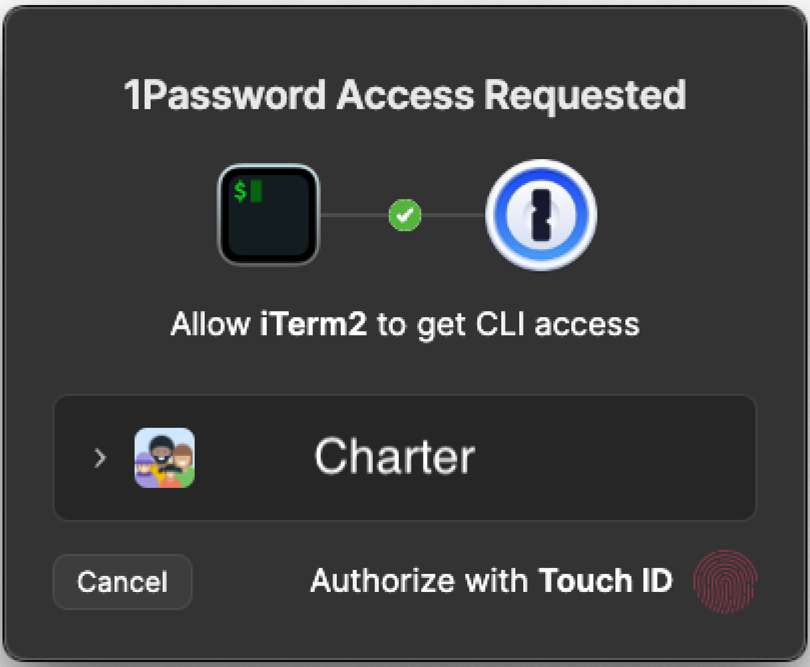
- Open a new terminal window or reinstantiate your shell with `exec $SHELL`
- Run `spoctunnel` to see help

```
❯ spoctunnel
spoctunnel (start|stop|tail)
```

- Run `spoctunnel start` to start the `sshuttle` application

```
❯ spoctunnel start
[4] 5073
```

- 1Password will prompt for fingerprint authentication. This is the **ONLY** authentication step required.


1Password Access Requested

Allow **iTerm2** to get CLI access

- Run `spoctunnel tail` to view logs

```
❯ spoctunnel tail
Starting sshuttle proxy (version 1.1.1).
c : Starting firewall manager with command: ['/usr/bin/sudo', '-p',
'[local sudo] Password: ', '/usr/bin/env',
'PYTHONPATH=/usr/local/Cellar/sshuttle/1.1.1/libexec/lib/python3.12/site-
packages', '/usr/local/Cellar/sshuttle/1.1.1/libexec/bin/python',
'/usr/local/bin/sshuttle', '-v', '--method', 'auto', '--firewall']
fw: Starting firewall with Python version 3.12.0
fw: ready method name pf.
c : IPv6 enabled: Using default IPv6 listen address ::1
c : Method: pf
c : IPv4: on
c : IPv6: on
c : UDP : off (not available with pf method)
c : DNS : on
c : User: off (not available with pf method)
c : Subnets to forward through remote host (type, IP, cidr mask width,
startPort, endPort):
c :   (<AddressFamily.AF_INET: 2>, '10.240.41.0', 22, 0, 0)
c :   (<AddressFamily.AF_INET: 2>, '10.240.12.0', 22, 0, 0)
c :   (<AddressFamily.AF_INET: 2>, '44.128.28.0', 24, 0, 0)
c :   (<AddressFamily.AF_INET: 2>, '44.114.100.0', 24, 0, 0)
c :   (<AddressFamily.AF_INET: 2>, '44.128.32.0', 22, 0, 0)
```

- Run `spoctunnel stop` to shut down the `sshuttle` application

```
❯ spoctunnel stop
5073
5152
5155
5156
[4]  + 5073 terminated  SSHPASS=$(op read op://Charter/charterlab-
spoc/password) bash -c  >> $LOGFILE
```