The VPN HOWTO

Table of Contents

The VPN HOWTO

Arpad Magosanyi <mag@bunuel.tii.matav.hu> v0.2,7 Aug1997

v0.3, 2001−12−01

Archived Document Notice: This document has been archived by the LDP because it does not apply to modern Linux systems. It is no longer being actively maintained.

---

1. Changes

The 'no controlling tty problem' −> −o 'BatchMode yes' by Zot O'Connor <zot@crl.com>

warning about kernel 2.0.30 by mag

---

2. Blurb

This is the Linux VPN howto, a collection of information on how to set up a Virtual Protected Network in Linux (and other unices in general).

2.1 Copyright

This document is part of the Linux Documentation Project. The copyright notice is the following:

2.2 Disclaimer

As usual: the author not responsible for any damage. For the correct wording, see the relevant part of the GNU GPL 0.1.1

2.3 Disclaimer

We are dealing with security: you are not safe if you haven't got good security policy, and other rather boring things.

2.4 Credits

Thanks to all of who has written the tools used.

Thanks to Zot O'Connor <zot@crl.com> for pointing out the "no controlling tty" problem, and it's solution.

2.5 State of this document

This is very preliminary. You should have thorough knowledge of administrating IP, at least some knowledge of firewalls, ppp and ssh. You should know them anyway if you want to set up a VPN. I just decided to write down my experiences not to forget them. There are possibly some security holes indeed. To be fair I've tried it on hosts configured as routers not firewalls, saying: It's simple from that point.

2.6 Related documentations

- The Linux Firewall−HOWTO /usr/doc/HOWTO/Firewall−HOWTO
- The Linux PPP−HOWTO /usr/doc/HOWTO/PPP−HOWTO.gz
- The ssh documentations /usr/doc/ssh/*
- The Linux Network Admins' Guide
- NIST Computer Security Special Publications http://csrc.ncsl.nist.gov/nistpubs/
- Firewall list (majordomo@greatcircle.com)

---

3. Introduction

As firewalls are in more and more widely use in internet and intranet security, the ability to do nice VPNs is important. Here are my experiences. Comments are welcome.

3.1 Naming conventions

I will use the terms "master firewall" and "slave firewall", though making a VPN has nothing to do with client−server architecture. I simply refer to them as the active and passive participants of the connection's setup. The host which is starts the setup will be referred as the master, and the passive participant will be the slave.

---

4. Doing it

4.1 Planning

Before you start to set up your system, you should know the networking details. I assume you have two firewalls protecting one intranet per firewall, and they are both connected to the internet. So now you should have two network interfaces (at least) per firewall. Take a sheet of paper, write down their IP addresses and network mask. You will need one more IP adresses per firewall for the VPN you want to do now. Those addresses should be outside of your existing subnets. I suggest using addresses from the "private" address ranges. They are the followings:

- 10.0.0.0 − 10.255.255.255
- 172.16.0.0 − 172.31.255.255
- 192.168.0.0 − 192.168.255.255

For the sake of example, here's a sample configuration: The two bastions are called fellini and polanski. They have one interface for the internet (−out), one for the intranet (−in), and one for the vpn (−vpn). The addresses and netmasks:

- fellini−out: 193.6.34.12 255.255.255.0
- fellini−in: 193.6.35.12 255.255.255.0
- fellini−vpn: 192.168.0.1 point−to−point
- polanski−out: 193.6.36.12 255.255.255.0
- polanski−in: 193.6.37.12 255.255.255.0
- polanski−vpn: 192.168.0.2 point−to−point

So we have the plan.

4.2 Gathering the tools

You will need a

- Linux firewall
- kernel
- very minimal configuration
- ipfwadm
- fwtk
- Tools for the VPN
- ssh
- pppd
- sudo
- pty−redir

Current versions:

- kernel: 2.0.29 Use a stable kernel, and it must be newer than 2.0.20, because the ping'o'death bug. At the time of writing 2.0.30 is the last "stable" kernel, but it has some bugs. If you want to have the fast and cool networking code introduced in it, try a prepatch. the 3rd is working for me nicely.
- base system: I prefer Debian. YMMV. You absolutely don't want to use any big packages, and you never even tought of using sendmail, of course. You also definitely don't want to enable telnet, ftp, and the 'r' commands (as usual in case of any other unix hosts).
- ipfwadm: I've used 2.3.0
- fwtk: I've used 1.3
- ssh: >= 1.2.20. There are problems with the underlying protocol in the older versions.
- pppd: I've used 2.2.0f for the tests, but I'm not sure if is it secure, this is why I turned the setuid bit off, and used sudo to launch it.
- sudo: 1.5.2 the newest I am aware of
- pty−redir: It is written by me. Try ftp://ftp.vein.hu/ssa/contrib/mag/pty−redir−0.1.tar.gz. Its version number is 0.1 now. Tell me it there is any problem with it.

4.3 Compile and install

Compile or otherwise install the gathered tools. Look at every one's documentation (and the firewall−howto) for details. Now we have the tools.

4.4 Configure the other subsystems

Configure your firewall rules, etc. You need to enable ssh traffic between the two firewll hosts. It means a connection to port 22 on the slave from the master. Start sshd on the slave and verify if you can login. This step is untested, please tell me your results.

4.5 Set up the accounts for the VPN

Create an account on the slave firewall use your favourite tool (e.g. vi, mkdir, chown, chmod) you might create an account on the master also, but I think you want to set up the connection at boot time, so your ordinary root account will do. Can anyone point out risks on using the root account on the master?

4.6 Generate an ssh key for your master account

Use the ssh−keygen program. Set empty password for the private key if you want to do automatic setup of the VPN.

4.7 Set up automatic ssh login for the slave account

Copy the newly generated public key in the slave account under .ssh/authorized_keys, and set up file permissions like the following:

```
drwx−−−−−− 2 slave slave 1024 Apr 7 23:49 ./
drwx−−−−−− 4 slave slave 1024 Apr 24 14:05 ../
−rwx−−−−−− 1 slave slave 328 Apr 7 03:04 authorized_keys
−rw−−−−−−− 1 slave slave 660 Apr 14 15:23 known_hosts
−rw−−−−−−− 1 slave slave 512 Apr 21 10:03 random_seed
```

The first row being ~slave/.ssh, and the second is ~slave.

4.8 Tighten ssh security on the bastions.

It means the followings on my setup in sshd_conf:

```
PermitRootLogin no
IgnoreRhosts yes
StrictModes yes
QuietMode no
FascistLogging yes
KeepAlive yes
RhostsAuthentication no
RhostsRSAAuthentication no
RSAAuthentication yes
PasswordAuthentication no
PermitEmptyPasswords no
```

Password authentication is turned off, so login is only possible with authorized keys. (You've turned off telnet and the 'r' commands of course).

4.9 Enable execution of ppp and route for both accounts.

As the master account is the root in my case, it has nothing to do. For the slave account, the following lines appear in /etc/sudoers:

```
Cmnd_Alias VPN=/usr/sbin/pppd,/usr/local/vpn/route
slave ALL=NOPASSWD: VPN
```

As you can see, I am using some scripts to set up ppp and the routing tables on the slave host.

4.10 Do the scripting

On the master host there is a full−blown init script I am using:

```
#! /bin/sh
# skeleton      example file to build /etc/init.d/ scripts.
#               This file should be used to construct scripts for /etc/init.d.
#
#               Written by Miquel van Smoorenburg <miquels@cistron.nl>.
#               Modified for Debian GNU/Linux
#               by Ian Murdock <imurdock@gnu.ai.mit.edu>.
#
# Version:      @(#)skeleton  1.6  11−Nov−1996  miquels@cistron.nl
#

PATH=/usr/local/sbin:/sbin:/bin:/usr/sbin:/usr/bin:/usr/bin/X11/:
PPPAPP=/home/slave/ppp
ROUTEAPP=/home/slave/route
PPPD=/usr/sbin/pppd
NAME=VPN
REDIR=/usr/local/bin/pty−redir
SSH=/usr/bin/ssh
MYPPPIP=192.168.0.1
TARGETIP=192.168.0.2
TARGETNET=193.6.37.0
MYNET=193.6.35.0
SLAVEWALL=polanski−out
SLAVEACC=slave

test −f $PPPD || exit 0

set −e

case "$1" in
  start)
        echo setting up vpn
        $REDIR $SSH −o 'Batchmode yes' −t −l $SLAVEACC $SLAVEWALL sudo $PPPAPP >/tmp/device
        TTYNAME=`cat /tmp/device`
echo tty is $TTYNAME
        sleep 10s
        if [ ! −z $TTYNAME ]
        then
        $PPPD $TTYNAME ${MYPPPIP}:${TARGETIP}
        else
```

```
                echo FAILED!
                logger "vpn setup failed"
        fi
        sleep 5s
        route add -net $TARGETNET gw $TARGETIP
        $SSH -o 'Batchmode yes' -l $SLAVEACC $SLAVEWALL sudo $ROUTEAPP
    ;;
  stop)
        ps -ax | grep "ssh -t -l $SLAVEACC " | grep -v grep | awk '{print $1}' | xargs kill
    ;;
  *)
    # echo "Usage: /etc/init.d/$NAME {start|stop|reload}"
    echo "Usage: /etc/init.d/$NAME {start|stop}"
    exit 1
    ;;
esac

exit 0
```

The slave uses one script for routing setup (/usr/local/vpn/route):

```
#!/bin/bash
/sbin/route add -net 193.6.35.0 gw 192.168.0.1
```

and its .ppprc consists of the following:

```
passive
```

5. Look at what's happening:

The master logs in into the slave, starts pppd, and redirects this all thing into a local pty. It consists of the following steps:

- allocating a new pty
- sshing into the slave
- running pppd on the slave
- the master runs pppd in this local pty
- and sets up the routing table on the client.

There are (not very tight) timing considerations involved, this is why that 'sleep 10s'.

6. Doing it by hand.

6.1 Logging in

You've already tried if ssh works well, aren't you? If the slave refuses to log you in, read the logs. Perhaps there are problems with file permissions or the sshd setup.

6.2 Firing up ppp

Log in into slave, and issue:

```
sudo /usr/sbin/pppd passive
```

You should see garbage coming at this point. If it works good, if not, there is some problem either with sudo, either with pppd. Look what the commands had said, and at the logs and at the */etc/ppp/options*, and the *.ppprc* file. If it works, write this 'passive' word into .ppprc, and try again. To get rid off the garbage and continue working, press enter,'~' and '^Z'. You should have the master's prompt now, and kill %1. See the section about tuning if you want to know more of the escape character.

6.3 Together the two

Well, then

```
ssh −l slave polanski sudo /usr/sbin/pppd
```

should work also, and deliver the garbage right into your face.

6.4 Pty redirecting

Try to redirect this whole thing this time:

```
/usr/local/bin/pty−redir /usr/bin/ssh −l slave polanski sudo /usr/sbin/pppd
```

Nice long sentence isn't it? You should use the full path into the ssh executable, as the pty−redir program allows only this form for security reasons. Now you've got a device name from the program. Let's say, you've got */dev/ttyp0* You can use the ps command to look what has happened. Look for 'p0'

6.5 Is anything on the device?

Try

```
/usr/sbin/pppd /dev/ttyp0 local 192.168.0.1:192.168.0.2
```

to establish the connection. Look at the output of the ifconfig command to see if the device has established, and use ping to check your virtual net.

6.6 Setting up the routes

Set up the routes on the master host, and on the slave also. Now you should be able to ping one host in one intranet from other host in the other intranet. Set up the additional firewalling rules. Now as you have the VPN, you can set up the rules concerning the connectivity of the two intranets.

7. Tuning

7.1 Configuration tuning

As I said this HOWTO is mainly a quick memo on how I had set up a VPN. There are things in the configuration I didn't experiment yet. These things will go into their place when I try them, or anyone tells me "it works in the following way" The most important thing is that the connection ppp uses is not 8−bit yet. I believe it has something to do either with ssh configuration or the pty setup. In this configuration ssh uses the tilde (~) character as an escape character. It might stop or slow down the communication, as any newline−tilde sequence causes ssh to give a prompt. Ssh documentation said: <On most systems, setting the escape character to ``none" will also make the session transparent even if a tty is used.> The corresponding flag to ssh is '−*e*', and you can also set it in the configuration file.

7.2 Bandwith vs. cicles

Creating anything virtual comes with utilization of real−world resources. A VPN eats up bandwidth and computing resources. The goal would be to get balance between the two. You can tune it with the '−C' switch or the 'CompressionLevel' option. You might try using another cipher, but I don't recommend it. Also note that the round−trip−time can be longer if you use better compression. Any experiments on it are welcome.

---

8. Vulnerability analisis

I try to cover here the vulnerability issues arising from this particular setup and VPNs in general. Any comments are warmly welcome.

- sudo: Well, I'm excessively using sudo. I believe it's still safer than using setuid bits. It's still a backdraw of Linux that it hasn't got more fine−grained access control. Waiting for POSIX.6 compatibility <http://www.xarius.demon.co.uk/software/posix6/>. What is worse, there are shell scripts which are getting called through sudo. Bad enough. Any idea out there?
- pppd: It runs suid root also. It can be configured by user's .ppprc. There might be some nice buffer overruns in it. The bottom line: secure your slave account as tightly as you can.
- ssh: Beware that ssh older than 1.2.20 has security holes. What is worse, we made a configuration such when the master account had been compromised, the slave account is also compromised, and wide open to attacks using the two sudoed programs. It is because I've choosen not to have password on the master's secret key to enable automatic setup of the VPN.
- firewall: With inproperly set firewall rules on one bastion, you open both of the intranets. I recommend using IP masquerading (as setting up incorrect routes is a bit less trivial), and doing hard control on the VPN interfaces.

8.1 About the ppp over ssh VPN technique

I developed this technique when there was no usable, standard VPN for Linux. Now this is no longer the case. At the time of writing this, you have the following alternatives: If you want to use standard IPSEC VPN, you can use FreeS/WAN or pipsecd. For PPTP you can use PoPToP (but be aware that PPTP protocol has weaknesses). It is also worth to mention CIPE which is a lightweight alternative for IPSEC.

This wide range of alternatives means that the ssh/ppp implementation described in this howto is in the most cases not the best solution. This is due the fact that this implementation is complex to set up and has performance problems because of its tcp based nature.

I believe that the ssh/ppp technique is no longer beneficial for building a VPN for non−illegal purposes in most cases, so I have discontinued maintaining this HOWTO.