

ASUS WL-Series Webserver guide

This is a step for step guide in turning your ASUS WLxxxx linux-based wireless product, into a small webserver.

Following the guide will not change the default features of your device, it will just add some nice functionality to it.

The guide is made using a ASUS WL-500G Deluxe (or WL-500GX as it is sometimes called). It should however apply to all the ASUS wireless line of products that are Linux based. These products are to my knowledge :

- ASUS WL-500b
- ASUS WL-500bv2
- ASUS WL-500g
- ASUS WL-500g Deluxe (WL500-gx)
- ASUS WL-HDD

If you are the lucky owner of any of the above devices, you have come to the right place.

If you still dont have one of the above devices - dont hesitate, go buy one :-)

After following the steps in this guide, your ASUS WL device will have all its normal functionality,
and also contain a full featured PHP enabled webserver that can be accessed from the WAN (Internet).

Changelog

28-04-2005

- Switched from thttpdphp to php-thttpd
- Changed webroot to comply with Unslung standards
- Corrected small error in setting eXecutable flag (thanks rdk)

17-04-2005

- Initial version

Guide Index

1. [Prerequisites](#) - What is needed to get started.
2. [Basic Knowledge](#) - Basic information about the system, and the packages that are beeing installed
3. [Disk Setup](#) - Setting up harddrive / usb memory key
4. [Installing basic software](#) - install a text editor and the webserver
5. [Configuring the webserver and ssh server](#)
6. [Making the webserver and the build-in ssh server startup at boot, and making them accessible from WAN](#)

Prerequisites

All you need is :

- One of the above Asus Wireless devices
- An USB Harddrive or USB Memory Stick / Keyring with some 50-100 MB space.(For the WL-HDD you just need a regular Harddrive)
- [Firmware version 1.9.2.7-4](#) by [Oleg](#) (and maby other versions, but tested on this one!)

The firmware IS needed. You will NOT get the steps below working if you use the stock firmware.

The firmware on your device can be upgraded using the webinterface, please consult your users manual
for details.

If you don't already own one of the ASUS devices above, but consider buying one in order to take advantage of the server-possibilities, you should probably go for the WL-500gx or the WL-HDD. The WL-500gx has USB2.0 and the WL-HDD uses a normal IDE interface, while the other devices use the slow USB 1.1 interface.

Basic Knowledge

The ASUS WL-series routers are all running on a Broadcom BCM947XX MIPSel CPU and a customized linux.

They all run a telnet server on the LAN side, allowing the users to access the routers console, using a telnet client like [Putty](#)

ASUS are not the only manufacturers making linux-based networking devices. Also [Linksys](#) have made quite a few products based on linux.

One of these products is the NSLU2, where a quite large [community](#) has developed a custom linux called [Unslung](#).

Unslung has its own package system, and a lot of these packages has been ported to the ASUS WL devices.

These packages are the ones used below.

Disk Setup

In the following, I will be using an USB disk, since I use the WL-500gx, not the WL-HDD. Even with the WL-HDD things should be the same.

I am not sure about the standard mount-point of the WL-HDD, but I guess it is the same as the other devices

There are multiple approaches regarding the disksystem. I have chosen to use a dedicated disk for storing my additional linux packages, and my html/php files for my webpages. Also I have chosen to use some of my disk space for SWAP, to ensure stability of my system.

Another approach would be to use only part of the disk for it, and then using the rest of it for the normal build-in ftpserver.

This can be done on two ways. One way is to use a FAT (windows) formatted disk, and create a file on that disk

that contains the ext3 filesystem. The other is to split the disk in multiple partitions, one with ext3, one with linux-swap and one with windows filesystem.

If there is demand for a walk-through of these methods, I will consider adding them to the guide.

The extra packages we are going to install are all going to be installed in the /opt directory on the system.

Since the /opt directory is empty and a part of the onboard Flash ram - which don't have enough space to store our files -

we are going to partition a disk, create the ext3 filesystem on it, and mounting it as /opt.

If you just connect a usb disk to the WL-500gx, it will automatically mount it to /tmp/harddisk and start the build-in ftp-server.

We need it to be mounted on /opt and hence need to make a few system-configuration changes.

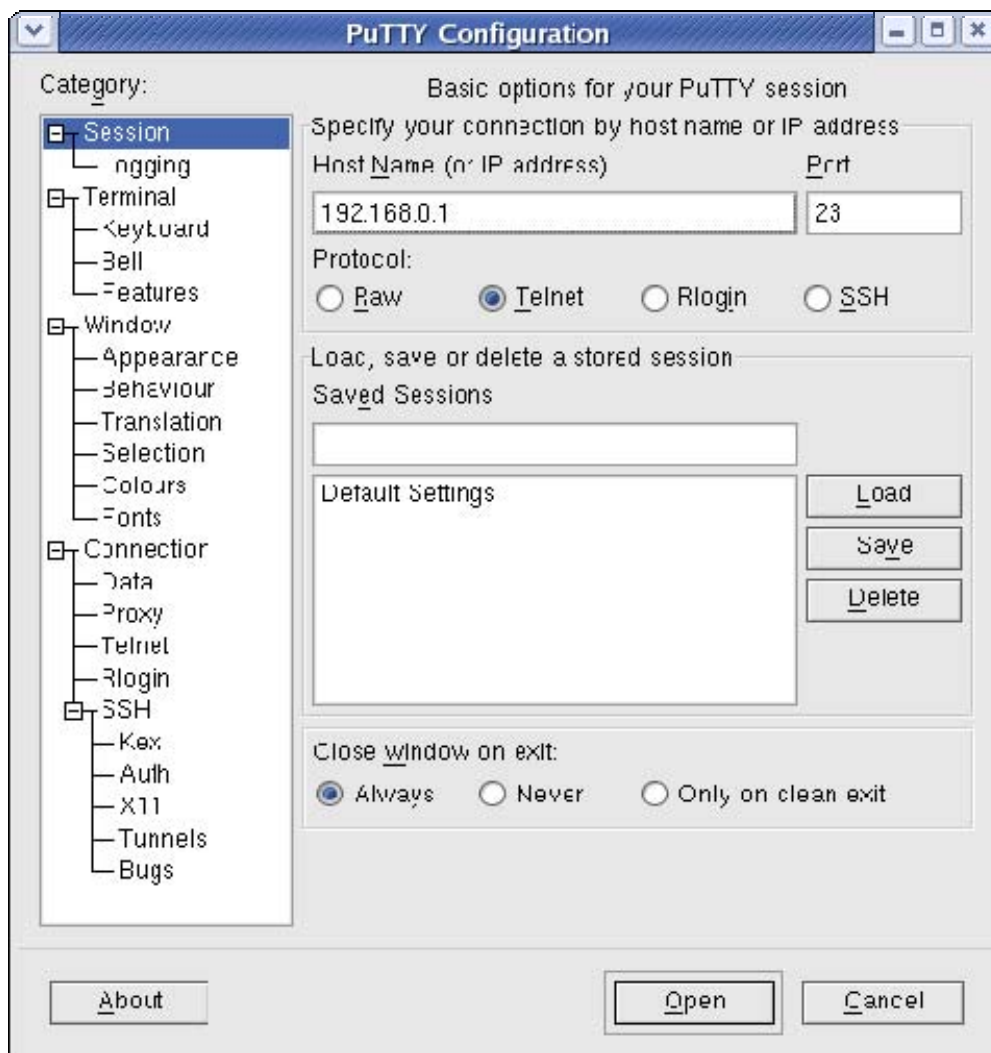
Configuring your disk space

1) Connect your USB memory/disk device (in this example I am using an USB2.0 256MB memory thing from Kingston (DataTravelerII))

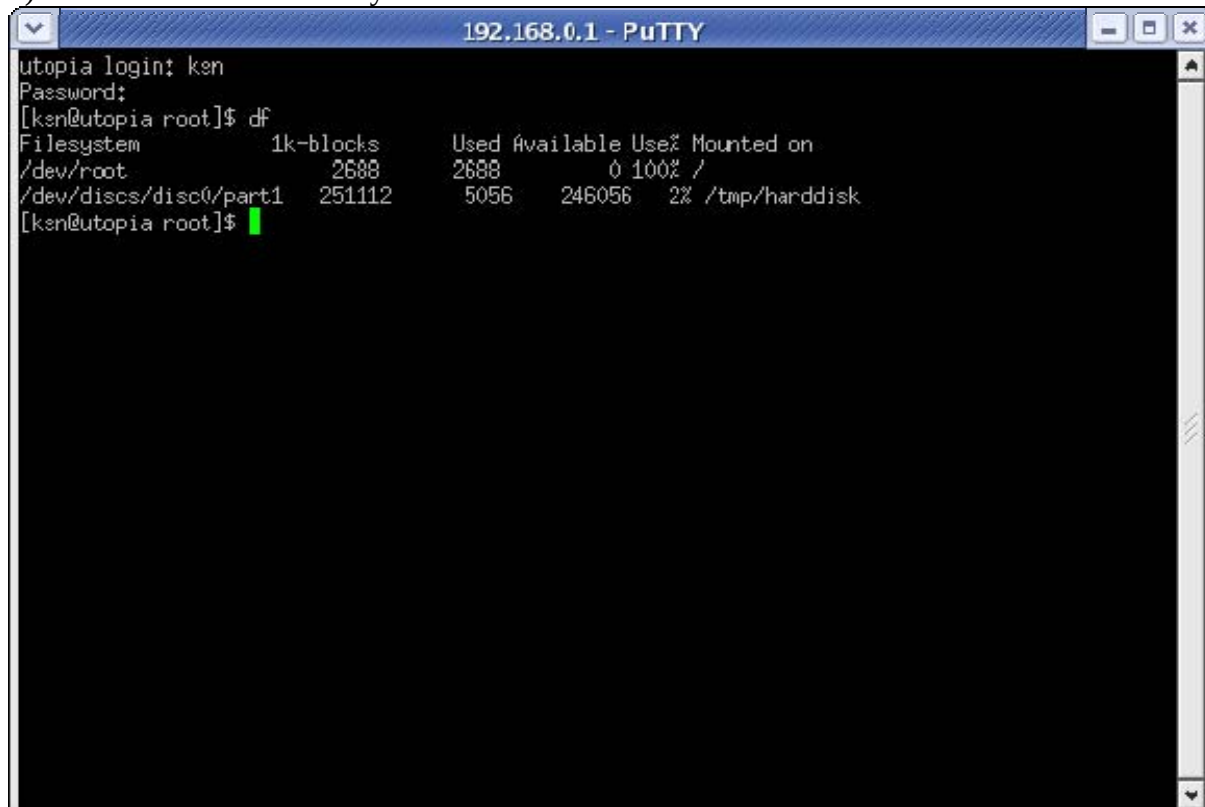
2) Log on to your router using telnet. As mentioned, [putty](#) would be a good choice of client.

If you haven't changed it, the router address is 192.168.1.1 and username / password is the same as you use for the webinterface.

(In my screenshot I use 192.168.0.1, since this is my router address!)



3) Check if the USB device you connected is mounted :

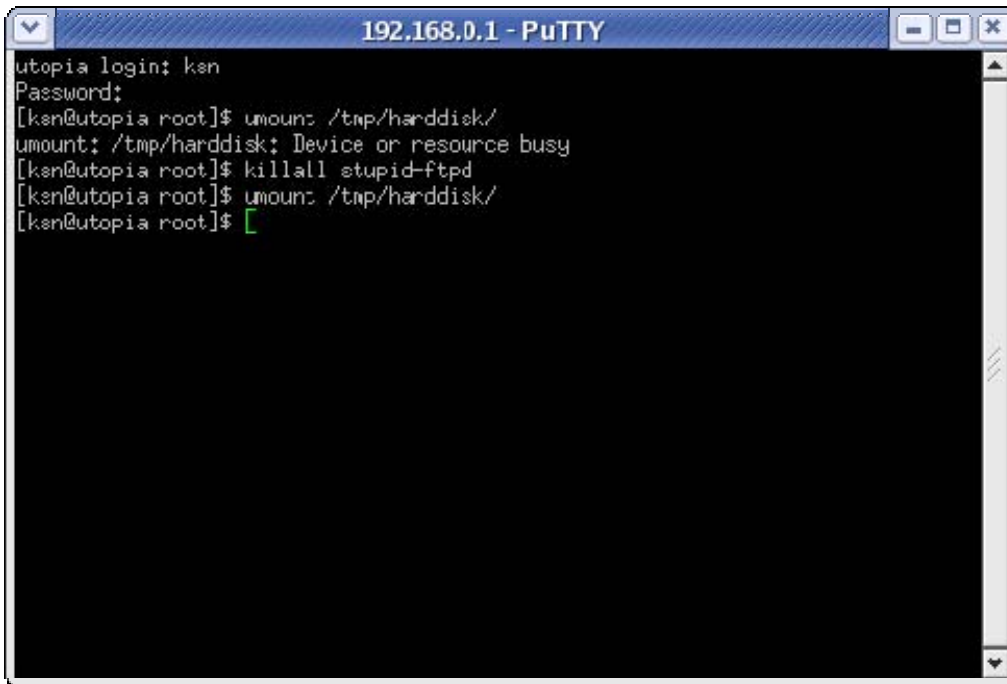


As you can see, my device automounted on : /tmp/harddisk and the devicename is : /dev/discs/disc0/part1 .

Whenever I write this devicename, you should replace with the devicename of YOUR device.

4) You need to unmount the device. If the umount command fails, it is probably because you have

enabled the ftpserver in your administration web pages. You can kill the ftp-server by issuing the commands as in this screenshot:



```
utopia login: ken
Password:
[ken@utopia root]$ umount /tmp/harddisk/
umount: /tmp/harddisk: Device or resource busy
[ken@utopia root]$ killall stupid-ftpd
[ken@utopia root]$ umount /tmp/harddisk/
[ken@utopia root]$
```

5) Now we are ready to create the linux (ext3) and swap partitions.

Issue the commands below, but please do READ what the output is as you move along.

Particularly you should consider changing values for "Last Cylinder / Size" to fit your harddrive size.

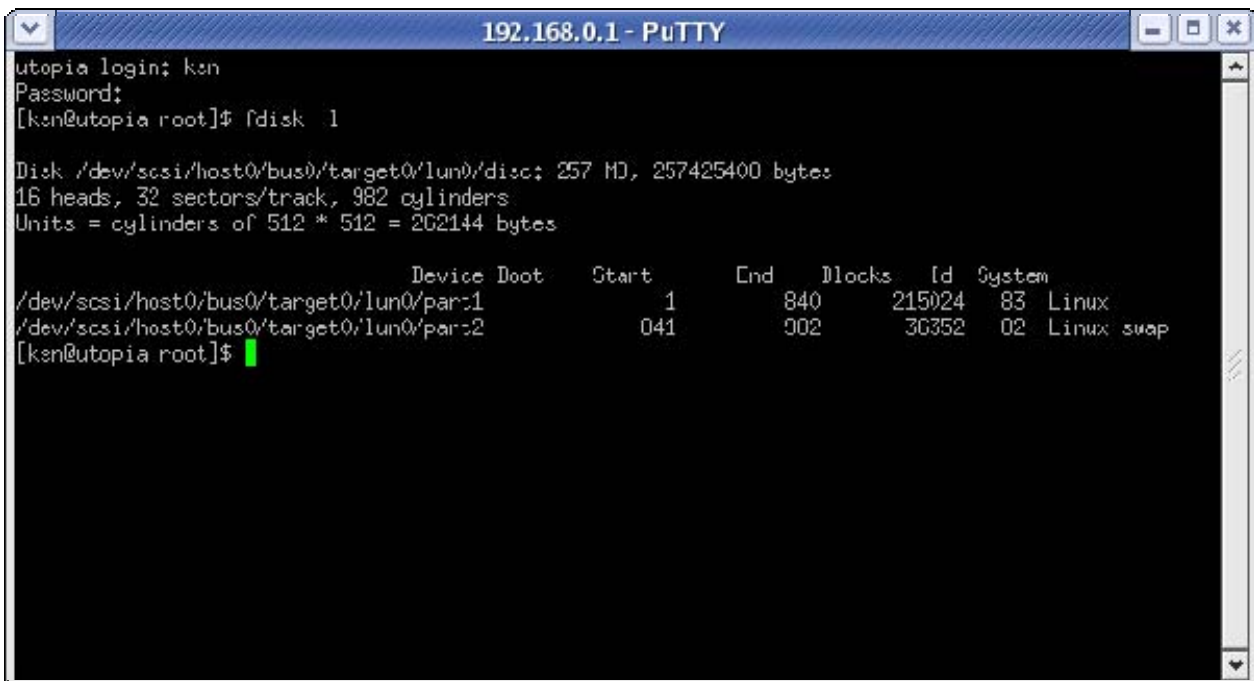
I am creating a 220 MB linux/ext3 partition, and a swap partition using the remaining space (36 MB).

```
fdisk /dev/discs/disc0/disc < enter >
d < enter >
n < enter >
p < enter >
1 < enter >
< enter >
+220M < enter >
n < enter >
p < enter >
2 < enter >
< enter >
< enter >
t < enter >
2 < enter >
82 < enter >
w < enter >
```

You can now list your new partitions by :

```
fdisk -l
```

My output is :



```
utopia login: ken
Password:
[ken@utopia root]$ fdisk /dev/sda

Disk /dev/sda: 257 MB, 257425400 bytes
16 heads, 32 sectors/track, 982 cylinders
Units = cylinders of 512 * 512 = 262144 bytes

   Device Boot      Start         End      Blocks   Id  System
/dev/sda1             1           840       215024    83  Linux
/dev/sda2             841          902        30352    02  Linux swap
[ken@utopia root]$
```

6) With the partitions created, filesystem has to be created (like when you are formatting your drives in windows).

```
mke2fs -j /dev/discs/disc0/part1 < enter >
mkswap /dev/discs/disc0/part2 < enter >
```

7) Now reboot your router, and after that login again using telnet, and if needed go through 4) again to unmount the device

```
reboot < enter >
```

8) Mount the newly created disk as /opt and enable the swap partition.

```
mount /dev/discs/disc0/part1 /opt < enter >
swapon /dev/discs/disc0/part2 < enter >
```

You can see your mounted drive, and the swap-space by :

```
df < enter >
free < enter >
```

Mount-on-boot will be enabled in the [Making the Webserver and build-in ssh server startup at boot](#) section of this guide.

Installing Basic Software

First thing to install is the ipkg package management system.

```
mkdir /opt/tmp < enter >
mkdir /opt/tmp/ipkg < enter >
ipkg.sh update < enter >
ipkg.sh install ipkg < enter >
```

Now update the packagelist for ipkg and install the [nano text editor](#) and the [httpd webserver](#) with php support.

```
/opt/bin/ipkg update < enter >
/opt/bin/ipkg install nano < enter >
/opt/bin/ipkg install openssl < enter >
/opt/bin/ipkg install zlib < enter >
/opt/bin/ipkg install php-thttpd < enter >
```

IF one of the commands return some text ended by: "Segmentation fault" try to issue the same command again.

It seems like the ipkg version is not completely stable.

Now you have successfully installed a nice editor, and a php enabled webserver

Configuring the webserver and ssh server

The webserver is quite easy to configure. First choose a place to store the html/php files, I have chosen /opt/www

```
mkdir /opt/share/ < enter >
mkdir /opt/share/www < enter >
```

Edit the webserver configuration file :

```
/opt/bin/nano /opt/etc/thttpd.conf < enter >
```

Change the file so that it contains :

```
dir=/opt/share/www
port=81
user=nobody
nochroot
nosymlink
novhost
logfile=/opt/var/log/thttpd.log
pidfile=/opt/var/run/thttpd.pid
```

Save the file by pressing < ctrl > + O followed by < enter >, exit nano with < ctrl > + X followed by < enter >

Create a couple of directories:

```
mkdir /opt/var < enter >
mkdir /opt/var/log < enter >
mkdir /opt/var/run < enter >
```

Now you can make a test-webpage by typing :

```
echo "This is my test webspace" > /opt/share/www/index.htm < enter >
```

To enable to ssh server, first generate a couple of keypairs for the ssl encryption:

```
mkdir -p /usr/local/etc/dropbear < enter >
dropbearkey -t dss -f /usr/local/etc/dropbear/dropbear_dss_host_key < enter >
dropbearkey -t rsa -f /usr/local/etc/dropbear/dropbear_rsa_host_key < enter >
```

It is important that you do NOT reboot your router, until you have finished the next section of this guide,

or at least done the "flashfs" command you can find below!

Now create the file /opt/etc/init.d/S13dropbear:

```
/opt/bin/nano /opt/etc/init.d/S13dropbear < enter >
```

Containing :

```
#!/bin/sh
dropbear
```

Save the file by pressing < ctrl > + O followed by < enter >, exit nano with < ctrl > + X followed by < enter >

Make the file executable :

```
chmod +x /opt/etc/init.d/S13dropbear < enter >
```

Making the webserver and build-in ssh server startup at boot, and making them accessible from WAN

The [Firmware version 1.9.2.7-4](#) by [Oleg](#) has a nice feature that it runs certain files at boot time, if they exist. Create these files :

```
mkdir /usr/local < enter >
mkdir /usr/local/sbin < enter >
touch /usr/local/sbin/post-boot < enter >
touch /usr/local/sbin/post-firewall < enter >
chmod +x /usr/local/sbin/* < enter >
```

Open the post-boot file in the nano editor:

```
/opt/bin/nano /usr/local/sbin/post-boot < enter >
```

Copy-Paste this into the editor window (on putty, paste with < enter > + < insert >):

```
#!/bin/sh
# wait for /opt to mount
mount /dev/discs/disc0/part1 /opt
i=0
while [ $i -le 30 ]
do
if [ -d /opt/etc ]
then
break
fi
sleep 1
i=`expr $i + 1`
done

# Activate swap
swapon /dev/discs/disc0/part2

# Run all active services - active means starts with S
/opt/etc/init.d/rc.unslung
```

Save the file by pressing < ctrl > + O followed by < enter >, exit nano with < ctrl > + X followed by < enter >

Now you need to open your firewall, to accept WAN (Internet) connections to your ssh and new www servers.

(these services are running on port 22 and port 80)

For doing this, you need to know which device your router uses for internet connection. This can be ppp0 or eth0 or

maybe even a few other things. To find out do

```
ifconfig < enter >
```

You will get something like:



```
[ksn@utopia sbin]$ ifconfig
br0    Link encap:Ethernet  HWaddr 00:11:D8:70:xx:xx
       inet addr:192.168.0.1  Bcast:192.168.0.255  Mask:255.255.255.0
       inet6 addr: fe80::211:d8ff:fe70:1f34/10 Scope:Link
       UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
       RX packets:26943 errors:0 dropped:0 overruns:0 frame:0
       TX packets:24254 errors:0 dropped:0 overruns:0 carrier:0
       collisions:0 txqueuelen:0
       RX bytes:4289852 (4.0 MiB)  TX bytes:9421283 (8.9 MiB)

eth0    Link encap:Ethernet  HWaddr 00:11:D8:70:xx:xx
       inet addr:83.91.xx.xx  Bcast:83.91.xx.xx  Mask:255.255.255.240
       inet6 addr: fe80::211:d8ff:fe70:1f34/10 Scope:Link
       UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
       RX packets:54874 errors:0 dropped:0 overruns:0 frame:0
       TX packets:51789 errors:0 dropped:0 overruns:0 carrier:0
       collisions:0 txqueuelen:100
       RX bytes:30227713 (28.8 MiB)  TX bytes:11138386 (10.6 MiB)
       Interrupt:4 Base address:0x1000

eth0.1  Link encap:Ethernet  HWaddr 00:11:D8:70:xx:xx
       inet6 addr: fe80::211:d8ff:fe70:1f34/10 Scope:Link
       UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
       RX packets:21612 errors:0 dropped:0 overruns:0 frame:0
       TX packets:23300 errors:0 dropped:0 overruns:0 carrier:0
       collisions:0 txqueuelen:0
       RX bytes:3671182 (3.5 MiB)  TX bytes:6412973 (6.1 MiB)

eth1    Link encap:Ethernet  HWaddr 00:11:D8:70:xx:xx
       inet6 addr: fe80::211:d8ff:fe70:1f34/10 Scope:Link
       UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
       RX packets:5329 errors:0 dropped:0 overruns:0 frame:11726
       TX packets:10026 errors:12 dropped:0 overruns:0 carrier:0
       collisions:0 txqueuelen:100
       RX bytes:779810 (761.5 KiB)  TX bytes:3713861 (3.5 MiB)
       Interrupt:2 Base address:0x4000

lo      Link encap:Local Loopback
       inet addr:127.0.0.1  Mask:255.0.0.0
       inet6 addr: ::1/128 Scope:Host
       UP LOOPBACK RUNNING MULTICAST  MTU:16436  Metric:1
       RX packets:10396 errors:0 dropped:0 overruns:0 frame:0
       TX packets:10396 errors:0 dropped:0 overruns:0 carrier:0
       collisions:0 txqueuelen:0
       RX bytes:888984 (868.1 KiB)  TX bytes:888984 (868.1 KiB)

[ksn@utopia sbin]$
```

The device you are interested in, is the one with inet addr: 83.91.xx.xx in my case, and something different in your case.

You can recognize it by it NOT being the same address as all the other devices. As you can see, in my case it is eth0.

All you need to do is to replace eth0 with your WAN device, and 192.168.0.1 with your LAN address in the following

and then paste this into post-firewall:

```
#!/bin/sh
iptables -D INPUT -j DROP
iptables -A INPUT -p tcp --dport 81 -j ACCEPT
iptables -A INPUT -p tcp --dport 80 -j ACCEPT
iptables -A INPUT -p tcp --dport 22 -j ACCEPT
iptables -t nat -A PREROUTING -i eth0 -p tcp --dport 80 -j DNAT --to-destination
192.168.0.1:81
```

```
iptables -t nat -A PREROUTING -i eth0 -p tcp --dport 22 -j DNAT --to-destination 192.168.0.1:22
```

```
iptables -A INPUT -j DROP
```

The above will open the firewall for web and ssh connections.

Only thing now, is to save the changes we made. Since /usr/local/sbin is not on the USB disk we connected,

but instead in the internal flash memory of the ASUS Device, we need to save it to flash and tell the router

to use the files by enabling flashfs :

```
flashfs save < enter >
flashfs commit < enter >
flashfs enable < enter >
```

That is basically it, reboot your ASUS WL device, and enjoy your ssh and web-server :-)

```
reboot < enter >
```

I hope you found this guide useful.

Comments are welcome to macsat@macsat.com

If you feel like it, donations are more than welcome as well :-)



Last modified: April 28 2005 18:51:31. CET