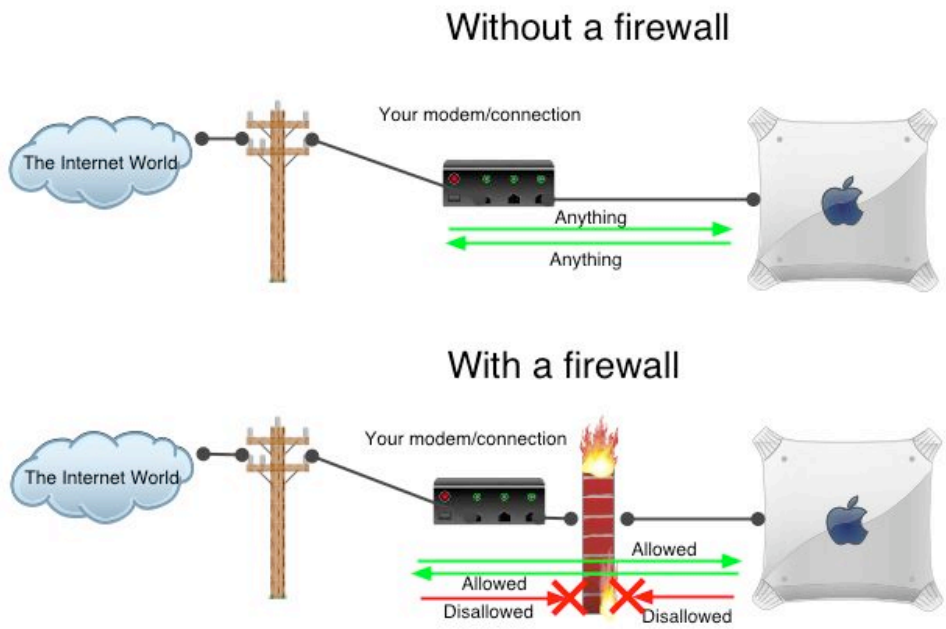
Setting up firewall rules on Mac OS X

Preamble

This page describes how I set up firewall rules on my Mac OS X (Jaguar or Panther) home computer using the built-in firewall (ipfw). You will need to use the Terminal application. If you want something simpler, go get the excellent **Brick House**, **Firewalk** or **SunShield** which all have simple options and a nice interface. In Mac OS X 10.2 and up, you can also use the System Preferences panel, Sharing: Firewall. For most cases (if you don't want anything fancy or if you don't know and don't want to know anything about firewalls), this is enough. If you use more than standard Apple services and want to know more, read on. Mom, stop here.

A firewall is a strict set of **rules** to allow or deny certain connections to or from your computer.
Without a firewall, any connection to your computer is allowed (see Figure below).



The firewall software is part of the operating system (i.e. you don't have to buy anything), and by default lets everything through (which means it is as if you had no firewall). Configuring your firewall means adding rules to permit only certain connections. The approach taken here is to explicitly allow only certain things to and from your computer, while blocking everything else. This is by far the most secure configuration.

All computers on the internet can communicate with each other if they have the required hardware (i.e. a network card, a modem, an airport card, etc...). These devices on your computer are related to a given **network interface**. These interfaces can be labelled with an acronym and a number that represent the language they talk (ppp0) or just with a generic name (en0). A network interface can also be logical (i.e. it is not related to an actual physical device) and play the role of routing traffic from one place to another. Network interfaces are the start and the end points of any connections.

But what is a **connection**? A connection is a discussion between two computers that are uniquely identified with an address (**IP address**, or IP for short) . The connection has a source IP address and a destination IP address. These refer respectively to the computer from which the connection was initiated and the destination computer to which the information is going.

To avoid various internet attacks that fake your IP address (or others') and fool your computer into doing something it should not do, you want to explicitly specify the address of your computer in the firewall rules. The idea is simple: there are certain IP addresses that represent something specific. For instance, the address 127.0.0.1 represents your computer, locally. Some other addresses are valid only when they come from "the inside world". Hence, by looking at the source IP address of an attempted connection, you can often prevent attacks to your computer. However, if you have a dynamic IP (often DSL, Cable), your IP address may change and you need to change the rules to reflect that. You could of course simply avoid using your IP address in the firewall rules, but it is not as safe as explicitly stating what your IP address is. The approach taken in this document is to be as restrictive as possible and always use your outside IP address. When the IP address changes, the firewall rules are reset.

A connection between two computers is made through specific **ports**. To make a simple analogy, the IP address is the building address and the port is the appartment number. Each computer (including yours) has several ports, numbered from 1 to 65000. Everything we do on the web (surfing the web, sending mail, etc...) is a use of a specific port on each computer. For instance, when you use a web browser, your computer connects to the port 80 of the web server through one of its own port. The ports below 1024 are considered *reserved ports* and are assigned to various **services** (http is port 80, ftp is 21, pop3 is 110, etc...). Your computer *listens* to some of the ports. For instance, if you have your own web server, your computer listens to port 80. If somebody tries to contact your computer via a web browser, it will do so by trying to connect to your port 80. But your computer also listens to some other ports, some of which should only be used by yourself (the port mechanism is very general and has more use than just communicating between two computers). Therefore, you want to restrict access to certain ports only and you do that with a firewall.

In addition to my regular internet connection (in my case, via DSL), I also have a tiny private network to share files with an iBook, which means I actually have two IP addresses: the outside IP (on interface ppp0, dynamic IP) and an internal static IP (inside interface en0, fixed IP 192.168.0.1). This configuration allows me to plug in an iBook at anytime at the back of my G4 and log in to kill jobs if I ever lose the console (not that I ever did, but it's nice to have it there). With the advent of Rendez-vous (Mac OS X 10.2), you don't even need that anymore: your computer can "create" IP addresses to be able to talk to other Rendez-vous enabled computers.

With a dynamic IP, you must make sure that the firewall is always synchronized with your IP. For instance, if you disconnect, then reconnect, your IP might have changed, but your firewall rules have not. You must then change the firewall rules to reflect the change of your IP. A script is available at the bottom of this page to do just that.

It might come to your attention when you set up the rules that your DNS Agent seems to time out pretty often, especially when you set up the rules. I don't know why that is.

ipfw Firewall rules for Mac OS X

I deserve no credit whatsoever since Peter Brezny at BSDToday has published an article on that.
The only thing I did is to make slight modifications:

- Turned on logging (kernel must be activated). Everything appears in /var/log/system.log
- Added a log for each deny (so you can trace who's trying to poke you).
- Added dynamic IP
- Removed the "divert natd" rule for those who don't run natd.

  If you do (for instance if you have a **second interface set up for Appletalk**, you must uncomment it). Most people will not require natd. The rule is simple: if you don't know what it is, you highly likely do not need it. If you do though (for instance if you share your internet connection amongst many computers), make sure you start natd before activating the firewall with the divert rule. If you don't you will be locked, simply start natd (which in these **instructions** is in /Library/StartupItems/NAT/) by typing /Library/StartupItems/NAT/NAT

Modify the file to suit your needs: outside interface (pppoe? ppp?) and inside interface (necessary?). I removed the original comments in and commented out the lines I modified. I found a few interesting **tutorials on Firewalls** at OnLamp.com if you want more info. Hopefully, my modifications are correct. Feel free to email me for **corrections**.

Download

You can get all the necessary files in the tar file **MacOS_X_Firewall.tar**. The tar file contains:

- The file rc.firewall.current, (the actual rules, see **below**)
- **Dynamic IP:** The files ip-up (to set up the firewall with a dynamic IP adresse i.e. modem, see **below**)
- **Static IP:** A file called Firewall and StartupParameters.plist which you copy into /Library/StartupItems/Firewall to start firewall at boot time if you have a static IP address (see **below**). Thanks to David Mackler for providing a better StartupItems.

The firewall rules

The rules are described below. To execute it, do `sh rc.firewall.current` as root, or `sudo sh rc.firewall.current`. Once you know it works, you can set up the computer to start the firewall **automatically**. The log is found at /var/log/system.log. If you make a mistake and it locks up, wait or type `ipfw flush`, look at the system.log and start over. You can look up every denied connection by typing `grep "Deny" /var/log/system.log`.

You need to modify the document in order to provide some information about your configuration, right after the line that says:
Define your variables.

- **oif**: your outside interface. This can be obtained with the following command in the terminal: `netstat -r | grep default`. The last word (pppoe0, en0, ppp0, etc) is your outside interface.
- **oip**: your outside ip address. If you have a dynamic IP address, leave the line as is to obtain it every time your run the script. This tiny script should do it. You can test it on the command line.
- **onwr**: your outside network range. $oip/8 is highly likely correct, but it could also be $oip/12 or $oip/16.
- **iif**, **inwr** and **iip**: your internal interface, internal network range and internal ip. If you do not have any, you should leave everything as is and it won't hurt.

Transition to 10.2 or 10.3 from 10.1: Mac OS X 10.2 (Jaguar) or 10.3 (Panther) has an udpated firewall software which allows what are called "dynamic rules". With those, it is possible to temporarily keep track of the friendly computers with which you have initiated connections and allow them back in when they talk to you. This is particularly useful for a certain type of connections (Domain Name server lookups) since they use UDP instead of TCP, and UDP connections have no short term memory (i.e. they are clueless, just like my boss). Also, Rendezvous makes use of certain addresses that used to be considered internal (draft-manning-dsua-01.txt below). We now allow those connections (I am not quite sure how safe that is, but that's all I can do for now). You will see (with sudo ipfw show) many dynamic rules to port 427 on your machine from machines like 239.255.255.253. That's normal, although I am unsure exactly what it is (svrloc?). Finally, the small script to extract the outside ip address now has a "-w" option to work properly
(thanks to Adrian Milliner for that).

```
#!/bin/sh
# Originally found at http://www.bsdtoday.com/2000/December/Features359.html
# By Peter Brezny
# Modifications done to support dynamic IP and default OS X configuration
# by D. Cote, available at: http://www.novajo.ca/firewall.html
#
# Simple stateful network firewall rules for IPFW with NAT v. 1.01
# See bottom of file for instructions and description of rules
# Created 20001206206 by Peter Brezny, pbrezny@purplecat.net (with a great
# deal of help from freebsd-security@freebsd.org).  Specific questions
# about the use of ipfw should be directed to freebsd-ipfw@freebsd.org or
# more general security questions to freebsd-security@freebsd.org.
# Use this script at your own risk.
#
# if you don't know the a.b.c.0/xx notation for ip networks the ipsubnet
# calculator can help you. /usr/ports/net/ipsc-0.4.2
#
############################
# Make sure logging is enabled (disabled by default)
if [ `/usr/sbin/sysctl -n net.inet.ip.fw.verbose` == 0 ] ; then
        /usr/sbin/sysctl -w net.inet.ip.fw.verbose=1
fi

#
# Define your variables
#
fwcmd="/sbin/ipfw"  # leave as is if using ipfw
oif="ppp0"          # set to outside interface name (for DSL pppoe0 in 10.0.x,
                    # pppoe0 in > 10.1.x)
                    # set following line to outside ip address
                    # or leave as is for dynamic IP address)
oip=`/sbin/ifconfig $oif| grep -w inet | awk '{ print $2 }'`;
onwr="$oip/8"           #set to outside network range
iif="en0"               #set to internal interface name
inwr="192.168.0.0/16"   #set to internal network range
iip="192.168.0.1"       #set to internal ip address


#
# End of required user input if you only intend to allow ssh connections to
# this box from the outside. If other services are required, edit line 96
# as necessary.
#
###
#   Rules with descriptions
#
#   Basic rules: there is no need to modify anything in this first section.
#   This is the bare minimum to block simple spoofing.
###
#
#   Force a flush of the current firewall rules before we reload
    $fwcmd -f flush

#   Allow your loop back to work
    $fwcmd add allow all from any to any via lo0

#   Prevent spoofing of your loopback
    $fwcmd add deny log all from any to 127.0.0.0/8

#
#   Stop spoofing of your internal network range
    $fwcmd add deny log ip from $inwr to any in via $oif
#
#   Stop spoofing from inside your private ip range
    $fwcmd add deny log ip from not $inwr to any in via $iif
#
#   Stop private networks (RFC1918) from entering the outside interface.
    $fwcmd add deny log ip from 192.168.0.0/16 to any in via $oif
    $fwcmd add deny log ip from 172.16.0.0/12 to any in via $oif
    $fwcmd add deny log ip from 10.0.0.0/8 to any in via $oif
    $fwcmd add deny log ip from any to 192.168.0.0/16 in via $oif
    $fwcmd add deny log ip from any to 172.16.0.0/12 in via $oif
    $fwcmd add deny log ip from any to 10.0.0.0/8 in via $oif
#
#   Stop draft-manning-dsua-01.txt nets on the outside interface
#   $fwcmd add deny log all from 0.0.0.0/8 to any in via $oif
#   $fwcmd add deny log all from 169.254.0.0/16 to any in via $oif
#   $fwcmd add deny log all from 192.0.2.0/24 to any in via $oif
#   $fwcmd add deny log all from 224.0.0.0/4 to any in via $oif
#   $fwcmd add deny log all from 240.0.0.0/4 to any in via $oif
#   $fwcmd add deny log all from any to 0.0.0.0/8 in via $oif
#   $fwcmd add deny log all from any to 169.254.0.0/16 in via $oif
#   $fwcmd add deny log all from any to 192.0.2.0/24 in via $oif
#   $fwcmd add deny log all from any to 224.0.0.0/4 in via $oif
#   $fwcmd add deny log all from any to 240.0.0.0/4 in via $oif
```

```
#
###
#   User rules: Some of the rules below are dependent on your configuration.
#   They might require some adjustments. They are emphasized with the
#   word "ADJUST".
###
#   ADJUST: If you use NATD (for your 192.168.0.1 interface for instance)
#   you must uncomment the following.  If you don't or if you don't know,
#   make sure next rule (divert) is commented.
#   Divert all packets through natd
#   $fwcmd add divert natd all from any to any via $oif

#
#   Allow all established connections to persist (setup required
#   for new connections).
    $fwcmd add allow tcp from any to any established

#
#   ADJUST: Allow incoming requests to reach the various services.
#   To allow multiple services you may list them separated
#   by a coma, for example ...to $oip 22,25,110,80 setup
#   If you have an internal interface (e.g. if you do not run NATd)
#   uncomment the second line to enable AppleTalk on it.
    $fwcmd add allow tcp from any to $oip ftp,ssh,http,svrloc,afpovertcp setup
#   $fwcmd add allow tcp from $oip to $iip 548 setup

#
#   NOTE: you may have to change your client to passive or active mode
#   to get ftp to work once enabled, only ssh, ftp and appletalk enabled by default.
#   21: ftp            enabled by default
#   22: ssh            enabled by default
#   23: telnet
#   25: smtp
#   80: http           enabled by default
#   110: pop
#   143: imap
#   80: http
#   427: svrloc (?)
#   443: ssl
#   548: appleshare    enabled by default
#   2401: cvs
#   5900-5909: VNC server, screen 0 through 9
#   6669: Limewire
#
#
#   Allow icmp packets for diagnostic purposes (ping traceroute)
#   you may wish to leave commented out.
    $fwcmd add allow icmp from any to any
#
#   Allow required ICMP
    $fwcmd add allow icmp from any to any icmptypes 3,4,11,12

#   Politely and quickly rejects AUTH requests (e.g. email and ftp)
    $fwcmd add reset tcp from any to $oip 113
#
#   Checks packets against dynamic rule set below.
    $fwcmd add check-state
#
#   Allow any traffic from firewall ip to any going out the
#   external interface
    $fwcmd add allow ip from $oip to any keep-state out via $oif

#
#   Allow any traffic from local network to any passing through the
#   internal interface
    $fwcmd add allow ip from $inwr to any keep-state via $iif

#
#   Deny everything else
    $fwcmd add 65435 deny log ip from any to any
#
#####################################################
#
# End firewall script.
```

Installing and starting the firewall

You need either option 1) or option 2), not both.

1) Dynamic IP address with ppp connection (most modems)

There used to be a shell script that would monitor the IP address and adjust the firewall accordingly when required. This broke with 10.1.x. However, I found out in the mean time that there is a much simpler way of doing that. With a ppp connection (i.e. modem), two scripts are automatically called when the connection is up and when the connection is taken down. They are /etc/ppp/ip-up and /etc/ppp/ip-down (see man pppd for more info in the terminal). Hence, one simply has to start the firewall in /etc/ppp/ip-up and flush the firewall in /etc/ppp/ip-down. I have provided two very simple ip-up and ip-down scripts. It assumes you will keep rc.firewall.current in /usr/local/sbin/. The script ip-up also works around a small bug in natd for 10.1 where it does not reset itself after a change of IP address (look at ip-up for more info, but it is pretty trivial).

Dynamic IP with ppp: start firewall when connection is up

```
cd /wherever/you/downloaded/MacOS_X_Firewall

sudo install -o root -g admin -m 0700 rc.firewall.current /usr/local/sbin/
sudo install -o root -g admin -m 0770 ip-up ip-down /etc/ppp/
```

You do not need anything in the /Library/StartupItems/ directory and if you do have a /Library/StartupItems/Firewall directory, you should delete it. You are done.

2) Static IP address without ppp

The best thing to do with a static IP address is to set up the firewall at boot time by adding an item in /Library/StartupItems/Firewall/. In the archive provided **above**, you will find Firewall and StartupParameters.plist which you will copy into /Library/StartupItems/. It assumes you will keep rc.firewall.current in /usr/local/sbin/.

Static IP and no ppp: start firewall on startup

```
cd /wherever/you/downloaded/MacOS_X_Firewall

sudo install -o root -g admin -m 0700 rc.firewall.current /usr/local/sbin/
sudo install -o root -g admin -m 0770 Firewall StartupParameters.plist /Library/StartupItems/Firewall/
```

Next time you reboot, the firewall will be up. For now, you should simply do: `sudo /Library/StartupItems/Firewall/Firewall`. No need to reboot, you are done.

Connection log

By default, if you don't change anything, every logged connection (i.e. those with a "log" option) will be written to /var/log/system.log. With Mac OS X 10.3, connections now get logged as "ipfw" and can be sent to a separate file. All you need to do is modifiy the file called /etc/syslog.conf.

Add this at the top of /etc/syslog.conf:

```
# Exclude ipfw entries from other logs
!-ipfw
```

Add this at the bottom:

```
# Redirect ipfw entries to ipfw.log
!ipfw
*.*                             /var/log/ipfw.log
```

Then call sudo kill -HUP `cat /var/run/syslog.pid`. Now all logged connections (denied or accepted) will be sent to /var/log/ipfw.log instead of system.log.

Troubleshooting

There are a few problems when you have a firewall:

- Always check /var/log/system.log (or /var/log/ipfw.log) for denied connections if you have problems. Do `grep "Deny" /var/log/system.log`. The explanation is often there.
- Is the firewall in sync with the IP address? In doubt, just run `sudo sh /usr/local/sbin/rc.firewall.current` again
- Is NATd in sync with the firewall (if you use it) ? Check the IP address used in the divert rule `sudo ipfw show | grep "divert"` and the system.log. Run `sudo sh /usr/local/sbin/rc.firewall.current` to correct the problem. Mac OS X 10.1 had a problem where NATd did not restart if the connection when brought down and then back up. I added a line to reset natd if it is running (in ip-up)
- Is NAT running ? If you have the divert rule set and NAT isn't running, you must start NAT
- If you use NAT and you do not have the divert rule activated, then NAT will never see the packets.
- ftp can give you trouble with a firewall if the FTP transfer is done in **active** mode instead of **passive** mode. An FTP connection is done on various ports > 49152. If the FTP server you are connectiong to is in **active** mode, the server will initiate the connection. With the firewall in place, this will be seen as an unauthorized incoming connection to your computer and will be rejected. On the other hand, if the FTP transfer mode is **passive**, your client will initiate the connection and since any outgoing connection is allowed, you will connect without a problem to the FTP server. Once the connection is established (i.e. the SYN bit of the TCP packet is set), connections are always allowed. Not every server supports it, but most do. Therefore, you might see (in your system.log) a server you know trying to connect from its port 21 to one of your ports >49152 until it times out. You must issue a **passive** command or configure your ftp software to do so (Interarchy , Fetch and Transmit all have an option for that: uncheck use **passive**; the command-line utility ftp usually defaults to active). I had some problems with Internet Explorer and I ended up changing the ftp helper application to Fetch. If you are not too paranoid, you can always just flush the firewall rules for the time of your download with the command `ipfw flush`. Any time you have a problem with your connection, go take a look at your system logs. If the connection is legitimate, modify the firewall rules to allow it.
- You might see denied connections on port 113 (authentication) and experience long delays when connecting to an FTP server or checking your mail. Some mail servers or FTP servers try to query your computer when you connect. The Authentification connection must time out before the real connection can go on. I added a rule to allow the mail server to connect to 113, but you don't really want to do that for every FTP server on the planet. You can explicitly reject connections to port 113 (using reset is better, thanks Bob K).

Contact information

You can email me at **dccote@novajo.ca** for corrections, comments or questions.